The Greedy Algorithm

Independence systems

A hypergraph $(E, \mathcal{F})$, $\mathcal{F} \subseteq 2^E$ is called an *independence system* if

(M1) $\emptyset \in \mathcal{F}$;

(M2) If $X \subseteq Y \in \mathcal{F}$, then $X \in \mathcal{F}$.

Sets in $\mathcal{F}$ are called *independent*, while those in $2^E \setminus \mathcal{F}$ are called *dependent*. Maximal independent sets are called *bases*, while minimal dependent sets are called *circuits*. The set of bases of $\mathcal{F}$ are denoted by $\mathcal{B} \subseteq \mathcal{F}$.

Many typical optimization problems can be formulated in the following way.

Maximum weight independent set problem
Given an independence system $(E, \mathcal{F})$ and real weights $c : E \mapsto \mathbb{R}_+$, find a subset $F \in \mathcal{F}$ such that $c(F) = \sum_{e \in F} c(e)$ is maximum.

Typical examples include the following:

Maximum weight stable set problem: Given a hypergraph $\mathcal{H} \subseteq 2^V$ for some set V, $|V| = n$, and real weights $c : V \mapsto \mathbb{R}_+$, let us call a subset $S \subseteq V$ *independent* if $H \not\subseteq S$ for all $H \in \mathcal{H}$. Let us denote by $\mathcal{F}$ the family of all independent sets for $\mathcal{H}$, and let $c(S) = \sum_{v \in S} c(v)$ for $S \in \mathcal{F}$. This formulation includes maximum stable set problems and maximum clique problems in graphs, etc.

Maximum weight matching problem: Given a graph $G = (V, E)$ and weights $c : E \mapsto \mathbb{R}_+$, let us call a subset $M \subseteq E$ of the edges *independent* if no two edges in M share a common endpoint (called a *matching* in G). Let $\mathcal{M}$ denote the family of all independent sets (matchings) of G and let $c(M) = \sum_{e \in F} c(e)$ for $M \in \mathcal{M}$.

Minimum weight perfect matching problem: Given a graph $G = (V, E)$, a matching $M \subseteq E$ is called a *perfect matching* if every vertex of G is incident with one of the edges of M (in which case $|V|$ must be an even number). Let us call a subset $F \subseteq E$ of the edges *independent* if $E \setminus F$

contains a perfect matching, and denote by $\mathcal{P} \subseteq 2^E$ the family of all independent sets. Given a weight $c \in \mathbb{R}_+^E$ the problem of finding an independent set $S \in \mathcal{P}$ such that $c(S) = \sum_{e \in S} c_e$ is as large as possible is equivalent with finding a perfect matching M for which its weight $c(M) = \sum_{e \in M} c_e$ is as small as possible.

Maximum weight spanning tree/forest problem: Given graph $G = (V, E)$ and weights $c : E \mapsto \mathbb{R}_+$, let us call a subset $T \subseteq E$ of the edges *independent* if it contains no cycles (such cycle free subgraphs are also called forests of G; if it is a connected spanning cycle free edge set, then it is called a tree of G). Let $\mathcal{T}$ denote the family of all independent sets (forests) of G and let $c(T) = \sum_{e \in F} c(e)$ for $T \in \mathcal{T}$.

Shortest path problem: Given a graph $G = (V, E)$, vertices $s, t \in V$, and weights $c : E \mapsto \mathbb{R}_+$, let us call a subset $Q \subseteq E$ of the edges *independent* if $(V, E \setminus Q)$ contains an $s - t$ path. Let $\mathcal{Q}$ denote the family of all independent sets (complements of $s - t$ connected subgraphs) of G and let $c(Q) = \sum_{e \in F} c(e)$ be the weight of an independent set $Q \in \mathcal{Q}$. Then finding a maximum weight independent set is equivalent with finding a path $P \subseteq E$ that connects s with t and has minimum weight $c(P) = \sum_{e \in P} c(e)$.

Generalized knapsack problem: Thinks of n favorite food items. Each item is described by a vector of its ingredients (like calories, A-vitamin content, D-vitamin content, cholesterol content, sugar content, etc.). For each ingredient we have a daily limit what can be taken safely. Call a subset S of the n food items independent if we can eat all items in S without violating our daily limits. Let us now denote by $c(i)$ the pleasure value of food item i, if we eat it. Then the maximum weight independent set problem is to find the subset of these food items that we can eat safely (without violating our daily limits) and achieving maximum level of culinary pleasures.

Set covering problem/hitting set problem: Think now of the beneficial nutrient groups, as sets of food items that you can eat, such as proteins, fibers, A-vitamin, D-vitamin, etc. The nutritional requirement is that you need to eat some food with each of these contents every day. Consider now n food items such that each is characterized by $0 - 1$ components, 1 indicating that it fulfills a particular nutritional group requirement (has enough fibers, or has enough proteins, etc.) while 0 indicates that it does not. If we know the cost of all food items, then we may want to find the subset of food items that fulfill the nutritional requirements, and has minimum total cost. This is called minimum hitting set problem. To approach this, call a set S of food items independent if eating all other food items that are NOT in S, you can still fulfill the nutritional requirements. The this is am independence system, and finding the maximum cost independent set is equivalent with finding the minimum cost nutritionally fulfilling set of food items.

Some of these problems are NP-hard, some others are polynomially solvable. For each of them the following simple procedure, called the *(best-in) greedy* algorithm, can be applied; it provides a solution that maybe reasonable, but may not be optimal.

```
(Best-In) Greedy

Input: An independence system $(E, \mathcal{F})$, and weights $c : E \longrightarrow \mathbb{R}$.

Initialize: Set $F^G = \emptyset$, and sort $E = \{e_1, e_2, ..., e_m\}$ such that
        $c(e_1) \geq c(e_2) \geq \cdots \geq c(e_m)$.

Main Loop: For $k = 1, ..., m$ do:
        If $F^G \cup \{e_k\} \in \mathcal{F}$ then $F^G = F^G \cup \{e_k\}$.

Output: $F^G$.
```

Complexity of Best-In Greedy: Sorting in **Initialization** takes $O(m \log m)$ time, while **Main Loop** can be executed in $O(m)$ time with $O(m)$ calls to the independence oracle. Thus, the total time needed to run (Best-In) Greedy is $O(m \log m + mI)$, where I is the worst case time we need to run the independence oracle.

Correctness of Best-In Greedy: We did not promise too much, so it is easy to verify that we have $F^G \in \mathcal{F}$ upon termination, since in the **Main Loop** F^G is updated only if it remains independent.

However, F^G may not be of maximum weight! The above greedy algorithm is only a *heuristic*, aimed at finding quickly (in polynomial time) a "good" solution, but with no guarantees of optimality.

Still, in many cases it provides an optimal solution, and in many other cases we can obtain reasonable performance guarantees for it.

Analysis of Best-In Greedy

To a subset $X \subseteq E$ we associate the induced sub-hypergraph $\mathcal{F}_X = \{X \cap F \mid F \in \mathcal{F}\}$, and let us denote by $\mathcal{B}_X$ the set of bases of $\mathcal{F}_X$. Let us define the *rank* of a subset $X \subseteq E$ by

$$r(X) \;=\; \max_{F \in \mathcal{F}} |F \cap X| \;=\; \max_{B \in \mathcal{B}_X} |B|.$$

Let the *lower rank* of X be defined by

$$\rho(X) \;=\; \min_{B \in \mathcal{B}_X} |B|,$$

and finally let the *rank quotient* of $(E, \mathcal{F})$ be defined as

$$q(E, \mathcal{F}) \;=\; \min_{X \subseteq E} \frac{\rho(X)}{r(X)}.$$

Lemma 1 *With the above definitions, we have for all* $F \in \mathcal{F}$, $X \subseteq E$, *and* $B \in \mathcal{B}_X$ *that*

$$|B| \geq \rho(X) \tag{1}$$

$$\rho(X) \geq q(E, \mathcal{F}) r(X) \tag{2}$$

$$r(X) \geq |F \cap X| \tag{3}$$

$\square$

Theorem 1 (Jenkins (1976), Korte and Hausmann (1978)) *Given an independence system* $(E, \mathcal{F})$ *and weights* $c : E \mapsto \mathbb{R}_+$, *let* F^G *denote the solution obtained by the above greedy procedure, and let* F^{OPT} *denote the optimal solution. Then we have*

$$q(E, \mathcal{F}) \leq \frac{c(F^G)}{c(F^{OPT})} \leq 1.$$

Furthermore, for every independence system there exist weights such that the lower bound is attained in the above statement.

Proof. Let $E_j = \{e_1, ..., e_j\}$ for $j = 1, ..., m$, and set $E_0 = \emptyset$. Let further $G_j = F^G \cap E_j$ and $O_j = F^{OPT} \cap E_j$ for $j = 0, ..., m$. To simplify notation, let us finally introduce $c(e_{m+1}) = 0$.

Then the statement will follow from the following chain of relations:

$$\begin{aligned} c(F^G) \;&=\; c(G_m) = \sum_{j=1}^{m} (|G_j| - |G_{j-1}|)\, c(e_j) \\ &= \sum_{j=1}^{m} |G_j|\, (c(e_j) - c(e_{j+1})) \end{aligned}$$

by (1) of Lemma 1 we have $|G_j| \geq \rho(E_j)$, since $G_j \in \mathcal{B}_{E_j}$, implying

$$\geq \sum_{j=1}^{m} \rho(E_j)\, (c(e_j) - c(e_{j+1}))$$

applying next (2) of Lemma 1 we get

$$\geq q(E, \mathcal{F}) \sum_{j=1}^{m} r(E_j)\, (c(e_j) - c(e_{j+1}))$$

finally, by (3) of Lemma 1 it follows that

$$\begin{aligned} &\geq q(E, \mathcal{F}) \sum_{j=1}^{m} |O_j|\, (c(e_j) - c(e_{j+1})) \\ &= q(E, \mathcal{F}) \sum_{j=1}^{m} (|O_j| - |O_{j-1}|)\, c(e_j) \\ &= q(E, \mathcal{F}) c(O_m) \;=\; q(E, \mathcal{F}) c(F^{OPT}). \end{aligned}$$

To see that this inequality is sharp, let us consider $X \subseteq E$ and $B_1, B_2 \in \mathcal{B}_X$ for which

$$q(E, \mathcal{F}) = \frac{|B_1|}{|B_2|},$$

and let us define

$$c(e) = \begin{cases} 1 \text{ for } e \in X \\ 0 \text{ for } e \in E \setminus X, \end{cases}$$

and sort the elements of E such that $B_1 = \{e_1, ..., e_{|B_1|}\}$. Then we get $F^G = B_1$ by Best-In Greedy, while $F^{OPT} = B_2$ is the optimal solution. $\square$

[How difficult is to compute $q(E, \mathcal{F})$ for the above examples?]

Matroids

It is possible to characterize independence systems for which the greedy algorithm always finds an optimal solution (i.e., for which $q(E, \mathcal{F}) = 1$, by the above theorem).

An independence system $(E, \mathcal{F})$ is called a *matroid* if it also satisfies

(M3) If $X, Y \in \mathcal{F}$, $|X| > |Y|$, then there exists an element $x \in X \setminus Y$ such that $Y \cup \{x\} \in \mathcal{F}$.

Lemma 2 *For independence systems $(E, \mathcal{F})$ axiom (M3) is equivalent with*

(M3') For each $X \subseteq E$, all bases of X are of the same size. $\square$

Corollary 1 *An independence system $(E, \mathcal{F})$ is a matroid iff $q(E, \mathcal{F}) = 1$.* $\square$

Typical examples of matroids include the following:

Linear matroids: Given a matrix $A \in \mathbb{R}^{m \times n}$, let E be the set of column vectors of A, and let us call a subset $I \subseteq E$ *independent*, if the corresponding column vectors are independent (equivalently, if the submatrix formed by those column vectors has rank $|I|$). Let $\mathcal{F}$ be the collection of all independent (column) subsets.

Graphic matroid: Given an undirected graph $G = (V, E)$, let us call an edge set $F \subseteq E$ *independent*, if it does not contain a cycle. Let $\mathcal{F}$ be the family of all cycle free edge subsets.

Transversal matroids: Let $\mathcal{H} = \{H_1, H_2, ..., H_n\}$ be subsets of a finite set V, $|V| = n$. A subset $S = \{s_1, s_2, ..., s_k\} \subseteq V$ is called *independent* (a partial transversal, or partial system of representatives), if there are indices i_1, i_2, ..., i_k such that $s_j \in H_{i_j}$ for $j = 1, ..., k$. Let $\mathcal{F}$ be the family of all partial transversals.

Scheduling example: Consider a problem of processing a set E of n jobs on a single machine. Each jobs $j \in E$ has a given processing time $t_j = 1$, a due date d_j, and brings a profit of c_j, if its processing is finished before its due date. Let us call a subset $S \subseteq E$ of the jobs *independent*, if all the jobs in S can be processed, one after the other, such that all of them are finished on time (before their respective due dates). Let $\mathcal{F}$ be the collection of all independent job subsets.

Theorem 2 (Rado (1957), Gale (1968) and Edmonds (1971)) *An independence system $(E, \mathcal{F})$ is a matroid if and only if* Best-In Greedy *finds an optimal solution for all nonnegative weight functions $c : E \mapsto \mathbb{R}_+$.*

Proof. If $(E, \mathcal{F})$ is a matroid, then by Corollary 1 we have $q(E, \mathcal{F}) = 1$, and thus the claim follows by Theorem 1.

For the converse direction, let us assume that $(E, \mathcal{F})$ is an independence system for which Best-In Greedy finds the optimal solution for every weight function $c : E \mapsto \mathbb{R}_+$. Let us further assume that $(E, \mathcal{F})$ is not a matroid, i.e., there exist independent sets $A, B \in \mathcal{F}$, such that $|A| > |B|$ and $B \cup \{a\} \notin \mathcal{F}$ for all $a \in A \setminus B$. Let $k = |B|$ and let us consider the weight function

$$c(e) = \begin{cases} k+2 \text{ if } e \in B \\ k+1 \text{ if } e \in A \setminus B \\ 0 \qquad \text{if } e \in E \setminus (A \cup B) \end{cases}$$

Then Best-In Greedy finds F^G for which $B \subseteq F^G \subseteq B \cup (E \setminus A)$, and hence $c(F^G) = |B|(k+2) = k(k+2)$. On the other hand

$$c(F^{OPT}) \geq c(A) = |A \cap B|(k+2) + |A \setminus B|(k+1) \geq |A|(k+1) \geq (k+1)^2.$$

Thus, $c(F^{OPT}) > c(F^G)$ follows, contradicting our assumption, and thus concluding the proof of the theorem. □

Matroid polytope

For a subset $S \subseteq E$ and a vector $x \in \mathbb{R}^E$ let $x(S) = \sum_{j \in S} x_j$. Given an independence system $(E, \mathcal{F})$ let us associate to it the so called *independent set polytope* $P_{E,\mathcal{F}}$, defined by

$$P_{E,\mathcal{F}} = \left\{ x \in \mathbb{R}^E \;\middle|\; \begin{array}{rcll} x(S) & \leq & r(S) & \text{for all } S \subseteq E \\ x_e & \geq & 0 & \text{for all } e \in E \end{array} \right\}$$

where r is the rank function of $(E, \mathcal{F})$. The inequalities

$$x(S) \leq r(S)$$

are called *rank inequalities*.

If $(E, \mathcal{F})$ is a matroid, then $P_{E,\mathcal{F}}$ is also known as the *matroid polytope*.

Clearly, if $S \in \mathcal{F}$, and $x^S \in \{0,1\}^E$ is the characteristic vectors of S, then $x^S \in P_{E,\mathcal{F}}$. Edmonds (1970) in fact proved that $P_{E,\mathcal{F}}$ is the convex hull of the characteristic vectors of independent sets.

Let us consider the following linear programming problem

$$\max\{c^T x \mid x \in P_{E,\mathcal{F}}\},$$

and its dual

$$\min \left\{ \sum_{S \subseteq E} r(S) y_S \;\middle|\; \begin{array}{rcll} \sum\limits_{\substack{S \subseteq E \\ e \in S}} y_S & \geq & c(e) & \text{for all } e \in E \\ y_S & \geq & 0 & \text{for all } S \subseteq E \end{array} \right\}.$$

Theorem 3 (Edmonds (1970)) *Let $(E, \mathcal{F})$ be a matroid, r its rank function, and $c : E \mapsto \mathbb{R}_+$. Let us sort the elements of $E = \{e_1, ..., e_m\}$ such that $c(e_1) \geq c(e_2) \geq \cdots \geq c(e_m)$, and let $E_0 = \emptyset$ and $E_j = \{e_1, ..., e_j\}$ for $j = 1, ..., m$. Let us then define*

$$y_S = \begin{cases} c(e_i) - c(e_{i+1}) & \text{if } S = E_i, i = 1, ..., m-1 \\ c(e_m) & \text{if } S = E_m, \\ 0 & \text{otherwise.} \end{cases}$$

Let further

$$x_i = \begin{cases} 1 & \text{if } r(E_i) > r(E_{i-1}) \\ 0 & \text{otherwise.} \end{cases}$$

Then $x = (x_i \mid i = 1, ..., m)$ is a primal optimal solution and $y = (y_S \mid S \subseteq E)$ is a dual optimal solution.

Proof. The vector x is the characteristic vector of the set F^G generated by Best-In Greedy, implying that $x \in P_{E,\mathcal{F}}$. We have $y \geq 0$, by definition. To check dual feasibility of y, let us choose an element $e_i \in E$ arbitrarily. Then we have

$$\begin{aligned} \sum_{S \subseteq E, e_i \in S} y_S &= \sum_{j \geq i} y_{E_j} \\ &= [c(e_i) - c(e_{i+1})] + \cdots + [c(e_{m-1}) - c(e_m)] + c(e_m) = c(e_i). \end{aligned}$$

Thus, x is feasible in the primal problem, while y is feasible in the dual. To prove the statement it is enough to show that

$$\sum_{i=1}^{m} c(e_i)x_i = \sum_{S \subseteq E} y_S = \sum_{i=1}^{m} r(E_i) y_{E_i},$$

which follow readily by

$$\begin{aligned} \sum_{i=1}^{m} c(e_i)x_i &= \sum_{i=1}^{m} c(e_i)[r(E_i) - r(E_{i-1})] \\ &= c(e_m)r(E_m) + \sum_{i=1}^{m-1} [c(e_i) - c(e_{i+1})]r(E_i) \\ &= \sum_{i=1}^{m} r(E_i) y_{E_i}. \end{aligned}$$

□

Maximum Weight Spanning Tree

Given a connected graph $G = (V, E)$ and weights $w_{u,v} \in \mathbb{R}$, $(u, v) \in E$ on its edges, find a spanning tree $T \subseteq E$ for which $w(T) = \sum_{e \in T} w_e$ is as large as possible. (Can we say max here? Why does it exist?)

Algorithm 1 Kruskal's (1956) Algorithm

```
1: procedure KRUSKAL($G = (V, E)$, $w : E \to \mathbb{R}$)
2:     Sort $E = \{e_1, e_2, ..., e_m\}$ such that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.
3:     Set $T = \emptyset$.
4:     for $k = 1 \to n$ do
5:         if $T \cup \{e_k\}$ is cycle free then
6:             $T = T \cup \{e_k\}$.
7:         end if
8:     end for
9:     return $T$
10: end procedure
```

Claim 1 *Kruskal's algorithm outputs the maximum weight spanning tree of the input connected graph in $O(|E| \log |E|)$ time.*

Proof. See in class. $\square$

[What happens if G is not connected?]

Another algorithm was discovered much earlier by Jarnik (1930), and later independently by Prim (1957) (and also by Disjkstra (1959).)

Algorithm 2 Prim's (1957) Algorithm

```
1: procedure Prim($G = (V, E)$, $w : E \to \mathbb{R}$)
2:     Set $T = \emptyset$ and $U = \{w\}$, where $w \in V$ is an arbitrary vertex.
3:     while $V \neq U$ do
4:         Find $(u^*, v^*) = \arg\max\{w(u,v) \mid (u,v) \in E,\ u \in U,\ v \notin U\}$.
5:         Set $T = T \cup \{(u^*, v^*)\}$.
6:         Set $U = U \cup \{v^*\}$.
7:     end while
8:     return $T$
9: end procedure
```

Claim 2 *Prim's algorithm returns the maximum weight spanning tree in* $O(|E| + |V| \log |V|)$ *time.*

Proof. See in class. $\square$

Compare also with Borůvka's (1926) algorithm.