Greedy Approximation Algorithm for Facility Location*[1]

- Set cover has a greedy algorithm which gives an H_d-factor approximation where d is the size of the largest set. In a previous note, we saw this analysis using a charging argument. In this lecture note, we see a much more sophisticated algorithm and analysis for a problem called the *metric* (uncapacitated) facility location problem. The problem is fundamental, and we will see many different algorithms for this in the course.

- **Facility Location.** In the facility location problem, we are given a set of facilities F, a set of clients C. Facility $i \in F$ has a cost f_i of opening. It costs $d(i,j)$ to connect client j to facility i. Clients can only be connected to open facilities. The objective is to find a set of facilities to open and connect clients to open facilities minimizing the total cost. This problem is usually called the *uncapacitated* facility location problem or simply UFL, so as to distinguish it from the capacitated facility location problem where each facility has a capacity which bounds the number of clients it can serve. The capacitated problem is much harder and out of the scope of this class.

  If the connection costs form a metric, that is, satisfy the triangle inequality

  $$d(i,j) \leq d(i,j') + d(j',i') + d(i',j), \quad \forall i,i' \in F, \;\; j,j' \in C$$

  then the problem is called the metric UFL.

- **Greedy Algorithm.** What should a greedy algorithm for the facility location would even look like? Let's consider the decisions to be made. One is to open facilities, and the other is to assign clients to open facilities. Opening a facility i incurs an opening cost f_i, and connecting a subset $Y \subseteq C$ of clients to i costs $d(Y,i) := \sum_{j \in X} d(i,j)$. Since, at the end, we have to assign every client to some open facility, if we were to take a hint from the set-cover algorithm, we would like to start by figuring out a facility i and a subset of clients X such that $\frac{f_i + d(Y,i)}{|Y|}$ is minimized. And then continue till all clients are covered. Indeed, one can go ahead and analyze this algorithm, but we won't.

  We notice that as stated above, the algorithm has at least one glaring weakness : we may end up assigning clients to a open facility which is not closest to it. Indeed, suppose that when we opened first facility i that we assigned Y to i. When we open the second facility i', some clients in Y may actually prefer moving to i' since i' is closer. And thus, opening something in i' may lead to a further "drop" in cost. The algorithm we describe below will take this into account.

  More precisely, the algorithm will maintain $X \subseteq F$ of open facilities and D the set of "covered" clients, and we will maintain an assignment $\sigma : D \to X$ indicating where these clients are assigned. For every facility i, let $D' \subseteq D$ be the covered clients j which are closer to i than to $\sigma(j) \in X$ where they are currently assigned. Thus, if we open i, we will definitely get a drop in the connection costs due to clients in D'. To this end, we use $\delta(D,i) = \sum_{j \in D} \max(0, d(\sigma(j),j) - d(i,j))$ to denote this drop. In each greedy step, we choose a facility i and a non-empty subset of not-yet-covered clients Y, so as to minimize the total cost $f_i + d(Y,i) - \delta(D,i)$ divided by $|Y|$. A crucial observation here is that in one of these steps, we may in fact choose a facility $i \in X$, in which case we don't pay the

[1] *Lecture notes by Deeparnab Chakrabarty. Last modified : 8th Jan, 2022*
These have not gone through scrutiny and may contain errors. If you find any, or have any other comments, please email me at deeparnab@dartmouth.edu. Highly appreciated!

opening cost again and nor do we get any saving, but rather we only consider $d(Y, i)/|Y|$. It may seem counter-intuitive to why such a step may ever take place, but we let the reader ponder on this[2]. Here is the full description.

1: **procedure** GREEDY-UFL($F \cup C, d$):
2: $\triangleright$ *X denotes the set of facilities opened and D denotes the set of covered clients.*
3: $\triangleright$ *Each client in D is assigned a facility in X via assignment $\sigma : D \to X$.*
4: Initially $X, D \leftarrow \emptyset$, and $\sigma \leftarrow \perp$.
5: **while** $D \neq C$ **do**:
6: $\triangleright$ *For a facility i, let $D' \subseteq D$ be the set of clients $j \in D$ who are closer to i than their currently assigned facility, that is, $d(i, j) < d(\sigma(j), j)$.*
7: $\triangleright$ *Let $\delta(D, i) := \sum_{j \in D} \max(0, (d(\sigma(j), j) - d(i, j))$ denote the reduction in connection costs among clients in D if i is opened.*
8: Pick a facility i and a set of unassigned clients $Y \subseteq C \setminus D$ so as to minimize

$$\Phi(i, Y) := \frac{\mathbf{0}_{i \in X} \cdot f_i + \sum_{j \in Y} d(i, j) - \delta(D, i)}{|Y|}$$

where $\mathbf{0}_{i \in X}$ is 0 if $i \in X$, and equals 1 otherwise. $\triangleright$ *How much time does this step take?*
9: $\triangleright$ Note if $i \in X$, then $\delta(D, i) = 0$.
10: $X \leftarrow X \cup i$; $D \leftarrow D \cup Y$
11: For all $j \in Y \cup D'$, assign $\sigma(j) \leftarrow i$.

Exercise: *How will one efficiently implement Line 8? As written it seems like going over all subsets. Show how to implement this using a greedy algorithm, and argue why it is correct.*

Analysis

Theorem 1. GREEDY-UFL is a 2-approximation.

- Let's fix some notation. Let X^* be the set of facilities opened by the optimal algorithm. Let σ^* be the assignment of clients to X^* of the optimal solution. Given a client j, let $d^*_j := d(\sigma^*(j), j)$. We let $F^* = \sum_{i \in X^*} f_i$ and $C^* = \sum_{j \in C} d^*_j$. Note that $\mathtt{opt} = F^* + C^*$. Similarly, let X be the facilities opened by GREEDY-UFL, and let σ be the final assignment. Let $d_j := d(\sigma(j), j)$, and $F_{\mathsf{alg}} = \sum_{i \in X} f_i$, and $C_{\mathsf{alg}} = \sum_{j \in C} d_j$. Note that $\mathsf{alg} = F_{\mathsf{alg}} + C_{\mathsf{alg}}$.
- As in the set-cover problem, we apply the "charging trick". Fix a client j and consider the *first iteration* when it enters D. This occurs when we choose a facility i (which can already be in X) and a subset Y which contains j. Assign a charge

$$\alpha_j := \frac{\mathbf{0}_{i \in X} \cdot f_i + \sum_{j \in Y} d(i, j) - \delta(D, i)}{|Y|}$$

Note that j could be re-assigned later on to a different facility, but we do not modify α_j.

[2]This may occur because we de-assign clients.

- We now make a key observation that at the end of the algorithm, the sum of charges on the clients is precisely the algorithm's cost.

$$\mathsf{alg} = F_{\mathsf{alg}} + C_{\mathsf{alg}} = \sum_{j \in C} \alpha_j \tag{1}$$

To see this, at every iteration $|Y|$ clients get the same α_j, and they sum to the numerator. The sum of the numerators over all iterations is precisely the facility opening costs plus the final connection costs with $\delta(D, i)$'s taking care of the drop of connection costs an individual client may face along the run of the algorithm.

- Now fix *any* facility $i \in F$ and *any* subset $Z \subseteq C$ of clients. We now make the following key claim.

Lemma 1. For any facility i and any subset $Z \subseteq C$, $\sum_{j \in Z} \alpha_j \le f_i + \left(2 - \frac{1}{|Z|}\right) d(Z, i)$ where $d(Z, i) := \sum_{j \in Z} d(i, j)$.

Before we prove the lemma, let us see how it implies Theorem 1. Consider the facilities $i \in X^*$, and for each such i let Z_i be the subset of clients assigned to i in the optimum solution. Thus, $F^* = \sum_{i \in F^*} f_i$ and $C^* = \sum_{i \in X^*} d(Z_i, i)$. Applying Lemma 1 for (i, Z_i) for $i \in X^*$, we get

$$\underbrace{\sum_{i \in X^*} \sum_{j \in Z_i} \alpha_j}_{= \sum_{j \in C} \alpha_j = \mathsf{alg}} \le \underbrace{\sum_{i \in X^*} f_i}_{= F^*} + 2 \cdot \underbrace{\sum_{i \in X^*} \sum_{j \in Z_i} d(i, j)}_{= C^*}$$

Thus, $\mathsf{alg} \le F^* + 2C^* \le 2\mathtt{opt}$ giving us the proof of Theorem 1.

- *Proof of Lemma 1.* Let $k := |Z|$ and order these k clients in the *order* in which they first covered by the algorithm, or more precisely, in the order they first are added to D. For simplicity, we assume that each of these clients appear in D at distinct loops; we leave it to the reader to notice that not having this assumption makes the analysis below only better. We rename these clients in Z as $(1, 2, \ldots, k)$ in this order.

  Now consider the iteration of the algorithm where client $j \in Z$ is added to D. At this juncture, some facility i is chosen (note i could already be in X) along with a set of clients Y containing j. Let D be the set of covered clients just before this decision, and let σ be the assignment of these clients in D at that point. By the greediness of the algorithm, we can infer that $\alpha_j \le \Phi(i', Y')$ for any other facility i' and subset Y'. We now find some specific i's and Y's to obtain usable upper bounds on α_j.

  - Note that by the ordering, all $\ell < j$ are already in D. Thus one possible choice for the algorithm was to not open any new facility but rather choose $i' = \sigma(\ell)$ and $Y' = \{j\}$, that is, connect j to where ℓ is connected. This puts the following upper bound on α_j. (See Figure 1 left side for an illustration.)

    $$\forall \ell < j, \quad \alpha_j \le d(\sigma(\ell), j)$$

    Now, and in fact the only place, we use ***metric-ness*** of the problem. We upper bound $d(\sigma(\ell), j)$ by noting that j could travel to i then to ℓ and then to $\sigma(\ell)$. This gives $\alpha_j \le d(\sigma(\ell), \ell) + d(i, j) + d(i, \ell)$. Adding for all $\ell < j$, gives

    $$(j-1)\alpha_j \le \sum_{\ell < j} d(\sigma(\ell), \ell) + (j-1) d(i, j) + \sum_{\ell < j} d(i, \ell) \tag{2}$$

– Another possible choice of the algorithm is to add the facility $i' = i$ and the set $Y' := \{\ell : \ell \geq j\}$. This gives us the following upper bound on α_j

$$\alpha_j \leq \frac{\mathbf{0}_{i \in X} \cdot f_i + \sum_{\ell \geq j} d(i,\ell) - \delta(D,i)}{(k-j+1)}$$

We now use (a) $\mathbf{0}_{i \in X} \leq 1$, and (b) $\delta(D,i) \geq \sum_{\ell<j} (d(\sigma(\ell),\ell) - d(i,\ell))$. To see why (b), consider reassigning clients in $\{1,\ldots,j-1\}$ to i (see Figure 1 right side for an illustration). This could be sub-optimal, but we only need a *lower bound* on the drop. Substituting, and rearranging, we get

$$(k-j+1) \cdot \alpha_j \leq f_i + d(Z,i) - \sum_{\ell<j} d(\sigma(\ell),\ell) \tag{3}$$

Figure 1: *An illustration for the above upper bounds.*

- Now we are almost done and all that remains is arithmetic. We add (3) and (2) to get

$$\text{For all } j \in Z, \quad k\alpha_j \leq f_i + d(Z,i) + (j-1)d(i,j) + \sum_{\ell<j} d(i,\ell)$$

Summing this for all $j \in Z$ and dividing by $k = |Z|$, we get

$$\sum_{j \in Z} \alpha_j \leq f_i + d(Z,i) + \frac{\sum_{j=1}^{k}(j-1)d(i,j) + \sum_{i=1}^{k}\sum_{\ell<j} d(i,\ell)}{k}$$

It takes a slight stare, but note that the numerator of the fraction above is precisely $(k-1)d(Z,i)$, which proves the lemma.

Notes

The greedy algorithm described above is present in the paper [2] by Jain, Mahdian, Markakis, Saberi, and Vazirani. However, the description is couched differently as a primal-dual algorithm, something which we encounter in later lectures. The paper contains two algorithms, and the algorithm in this notes is essentially Algorithm 2. The above analysis is essentially present in Section 8 of the JACM paper.

Jain *et al.* [2] actually prove that the approximation factor of the above algorithm is ≤ 1.61. However, note that the above analysis an $(1,2)$-approximation in the sense the algorithm's cost pays at most twice the optimal connection cost plus once the optimal facility opening cost. Such asymmetry can be exploited via a "greedy augmentation" trick present in [1] to get a better factor. The current best approximation factor for UFL is 1.488 and is present in the paper [3] by Li. It is known that unless $P = NP$, the best approximation one could hope for is 1.463. The latter result is present in [4].

References

[1] M. Charikar and S. Guha. Improved Combinatorial Algorithms for the Facility Location and k-Median Problems. In *Proc., IEEE Symposium on Foundations of Computer Science (FOCS)*, 1999.

[2] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *Journal of the ACM*, 50(6):795–824, 2003.

[3] S. Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Information and Computation*, 222:45–58, 2013.

[4] Sudipto Guha and Samir Khuller. Greedy Strikes Back: Improved Facility Location Algorithms. *Journal of Algorithms*, 31(1):228–248, Apr. 1999.