Local Search Algorithms for Facility Location*[1]

- In the metric UFL problem, we are given a set F of facilities, a set C of clients, and a metric $d(\cdot,\cdot)$ in $F \cup C$. Each facility $i \in F$ has an opening cost f_i. The objective is to open $X \subseteq F$ and connect clients via assignment $\sigma : C \to X$ to nearest open facility, to minimize

$$\texttt{cost}(X) = \sum_{i \in X} f_i + \sum_{j \in C} d(\sigma(j), j) \tag{1}$$

- **Local Search for UFL.** The local search algorithm has three operations : open, close, and swap, and if none of these three steps improve the solution, it terminates.

> **procedure** UFL-LOCAL SEARCH(F, C, d):
> X be an arbitrary subset of facilities.
>   ▷ *Throughout* $\texttt{cost}(X)$ *is defined using* (1)
>   **while** true **do**:
>   (Open): If there exists $i \in F \setminus X$ such that $\texttt{cost}(X + i) < \texttt{cost}(X)$; $X \leftarrow X + i$.
>   (Close): If there exists $i \in X$ such that $\texttt{cost}(X - i) < \texttt{cost}(X)$; $X \leftarrow X - i$.
>   (Swap): If there exists $i \in X, i' \in F \setminus X$ such that $\texttt{cost}(X - i + i') < \texttt{cost}(X)$; $X \leftarrow X - i + i'$.
>   Otherwise, break

- ***Analysis.*** We prove the following theorem.

> **Theorem 1.** UFL-LOCAL SEARCH is a 3-approximation algorithm.

Let X be the set of facilities opened at the end of the above algorithm. Let $\sigma(j)$ denote the facility in X client j is connected to. Let $\Gamma(i)$ denote the set of clients connected to facility $i \in X$. Let X^* denote the set of facilities opened in the optimal solution. Let σ^* and Γ^* be defined similarly. Let $d_j := d(\sigma(j), j)$ and $d^*_j := d(\sigma^*(j), j)$ be the connection costs for client j in the algorithm and optimum solution, respectively. Let $F_{\mathsf{alg}} = \sum_{i \in X} f_i$, $C_{\mathsf{alg}} = \sum_{j \in C} d_j$. Similarly define F^* and C^*.

- **Bounding C_{alg}.** This is relatively straightforward. We know that *opening* any facility doesn't decrease cost. Note that if we did open a facility $i \in X^*$, we could've moved all clients in $\Gamma^*(i)$ to i. Since this doesn't decrease cost (see Figure 1 for an illustration), we get that

$$\forall i \in X^*; \qquad \sum_{j \in \Gamma^*(i)} d_j \le f_i + \sum_{j \in \Gamma^*(i)} d^*_j \tag{2}$$

Adding over all $i \in X^*$ we get, $\sum_{i \in X^*} \sum_{j \in \Gamma^*(i)} d_j \le \sum_{i \in X^*} f_i + \sum_{j \in \Gamma^*(i)} d^*_j$, that is,

$$C_{\mathsf{alg}} \le F^* + C^*$$

[1] *Lecture notes by Deeparnab Chakrabarty. Last modified : 9th January, 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!

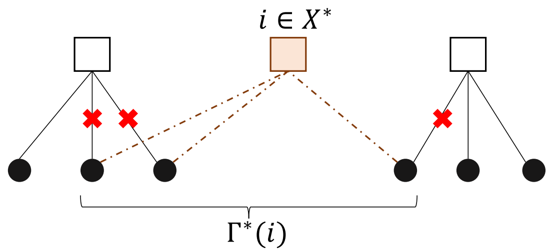

Figure 1: *Illustration of bounding C_{alg}.*

- **Bounding F_{alg}.** Fix an $i \in X$. How much can the connection cost of clients increase if i is deleted? All clients in $\Gamma(i)$ will move to their second-nearest facility in X. But what handle do we have on the distance between j and the second-nearest facility?

  Well we know, or at least have a handle on, the cost to connect j and $\sigma^*(j)$. It is d_j^*. So, if $\sigma^*(j)$ is in X, let's assign j to that. What if $\sigma^*(j)$ isn't there? Well, let's assign to the facility in X that is closest to $\sigma^*(j)$. This motivates the following key definition. See Figure 2 for an illustration

  Given $i^* \in X^*$, let $\texttt{nearest}(i^*)$ denote the facility i in X with minimum $d(i, i^*)$.

  Here is a useful fact which follows easily from triangle inequality and definition of `nearest` (see Figure 2 for an illustration).

  Claim 1. For any $j \in C$, $d(\texttt{nearest}(\sigma^*(j)), j) \leq d_j + 2d_j^*$.

  Proof. Let j be assigned to i in σ and i^* in σ^*. Then, triangle inequality implies $d(\texttt{nearest}(i^*), j) \leq d(i^*, j) + d(\texttt{nearest}(i^*), i^*) \leq d_j^* + d(i, i^*)$, where the last inequality is by definition of $\texttt{nearest}(i^*)$. Triangle inequality again implies $d(i^*, i) \leq d(i, j) + d(i^*, j)$. □

- Let's get back to our facility i. Let us look at clients $j \in \Gamma(i)$. If we close i, then we could reassign all these clients to $\texttt{nearest}(\sigma^*(j))$. The previous claim shows that this could increase the connection cost by at most $2d_j^*$ per client. By local optimality, since closing i doesn't decrease the total cost, we get that the facility opening cost of i must be at most $\sum_{j \in \Gamma(i)} 2d_j^*$. Which would then imply $F_{\mathsf{alg}} \leq 2C^*$, and then we would be done along with the bound on C_{alg}.

  Unfortunately, there is a fly in the ointment : what if $\texttt{nearest}(\sigma^*(j))$ is i itself for some $j \in \Gamma(i)$? Then, when i is closed, j can't be reassigned. To address this, we need to understand better how X^* and X behave w.r.t the `nearest` relation. This leads us to the next crucial definition.

- For any facility $i \in X$, define

$$X_i^* := \{i^* \in X^* : \texttt{nearest}(i^*) = i\}. \tag{3}$$

  that is, the facilities in X^* for which i is the closest facility. In some sense, it is the "inverse" of the `nearest` map, and indeed would exactly be that if `nearest` was a bijection. Instead, X_i^* maps to a subset of facilities in X^*. Crucially note that by definition, $X_i^* \cap X_{i'}^*$ for any two facilities in X.

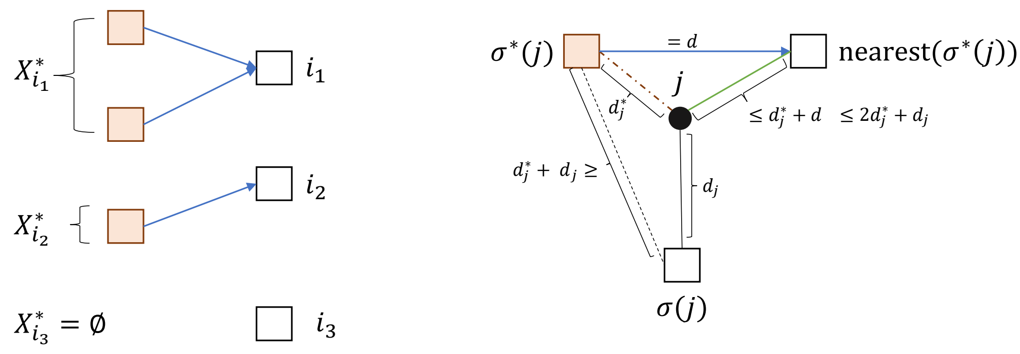

Figure 2: *Salmon squares denote facilities in X^* while empty squares denote facilities in X. The blue arrows denote the* `nearest` *map from X^* to X. The sets X^*_i for each $i \in X$ is denoted; note that $X^*_{i_1}$ has two facilities, $X^*_{i_2}$ has 1, while $X^*_{i_3}$ is empty. The right figure illustrates Claim 1.*

The following lemma bounds the facility opening cost of $i \in X$.

Lemma 1. For all $i \in X, \quad f_i \leq f(X^*_i) + \sum_{j \in \Gamma(i)} 2d^*_j$ where $f(X^*_i) := \sum_{i^* \in X^*_i} f_{i^*}$.

Before we prove the lemma, let us see that it implies the 3-approximation. Indeed,

$$F_{\mathsf{alg}} = \sum_{i \in X} f_i \leq \underbrace{\sum_{i \in X} f(X^*_i)}_{= F^* \text{ since } X^*_i\text{'s are disjoint and span } X^*.} + \sum_{i \in X} \sum_{j \in \Gamma(i)} 2d^*_j = F^* + 2C^*$$

Together with $C_{\mathsf{alg}} \leq F^* + C^*$, we complete the proof of Theorem 1.

- **Proof of Lemma 1.** The proof goes through three cases depending on the size of X^*_i.

- Case 0: $|X^*_i| = 0$. This is the case when there is no fly in the ointment. For every $j \in \Gamma(i)$, we vacuously have $\texttt{nearest}(\sigma^*(j)) \neq i$, and thus $\texttt{nearest}(\sigma^*(j)) \in X - i$. Now consider the local step of *closing* i, and reassigning all $j \in \Gamma(i)$ to $\texttt{nearest}(\sigma^*(j))$. Since this cannot lead to a decrease in the cost we get

$$f_i \leq \sum_{j \in \Gamma(i)} (\ d(\texttt{nearest}(j), j) - d(i,j)\ ) \underbrace{\leq}_{\text{Claim 1}} \sum_{j \in \Gamma(i)} 2d^*_j$$

proving the lemma in this case. The left figure in Figure 3 illustrates this case.

- Case 1: $|X^*_i| = 1$. Suppose $X^*_i = \{i^*\}$. In this case, consider *swapping* i and i^*. As in Case 0, we again have $\texttt{nearest}(\sigma^*(j)) \in X - i + i^*$ for all $j \in \Gamma(i)$. And since this swap doesn't help, we get

$$f_i \leq f_{i^*} + \sum_{j \in \Gamma(i)} (\ d(\texttt{nearest}(j), j) - d(i,j)\ ) \underbrace{\leq}_{\text{Claim 1}} f_{i^*} + \sum_{j \in \Gamma(i)} 2d^*_j$$

proving the lemma in this case. The right figure in Figure 3 illustrates this case.

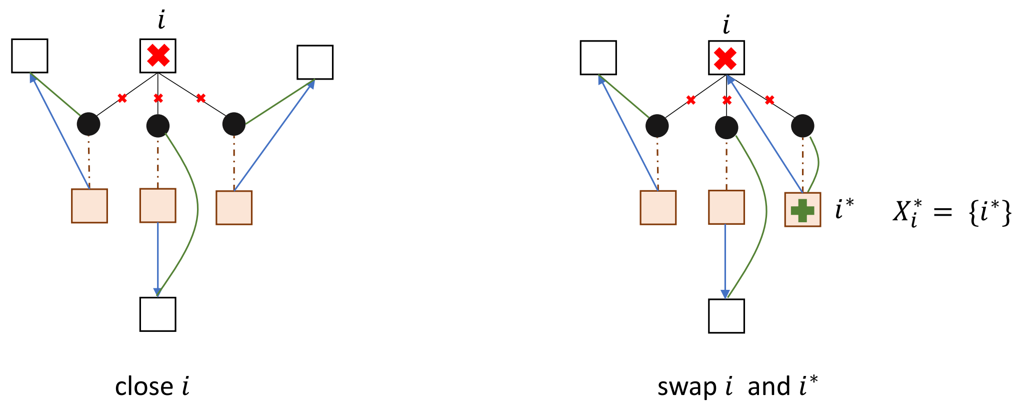

Figure 3: *Salmon squares denote facilities in X^* while empty squares denote facilities in X. Dotted brown lines denote the assignment σ^*. The blue arrows denote the* `nearest` *map from X^* to X. Green lines denote reassignments. In the figure on the left, $X_i^* = \emptyset$ and we just close i. For all $j \in \Gamma(i)$,* `nearest`$(\sigma^*(j))$ *points to a facility in $X \setminus i$ and they are reassigned there. In the figure on the right, $X_i^* = \{i^*\}$ and we swap i and i^*. For all $j \in \Gamma(i)$, if* `nearest`$(\sigma(j)) \neq i$ *then they are reassigned to that facility. If* `nearest`$(\sigma^*(j)) = i$*, then $\sigma^*(j)$ must be i^* in which case they are reassigned there.*

- Case 2: $|X_i^*| \geq 2$. This is a bit more interesting. Let's suppose $X_i^* = \{i_1^*, i_2^*, \ldots, i_k^*\}$ for some $k \geq 2$, and suppose we have ordered them in *increasing* order of distance from i. Next, we partition $\Gamma(i)$ into $k+1$ sets depending on where they go in the optimal solution, as follows.

$$A_0 := \{j \in \Gamma(i) \ : \ \sigma^*(j) \notin X_i^*\}; \quad \forall 1 \leq t \leq k, \quad A_t := \{j \in \Gamma(i) \ : \ \sigma^*(j) = i_t^*\}$$

Now, as in Case 1, consider *swapping* i and i_1^*. See the left figure in Figure 4 for an illustration. Note that clients $j \in A_0 \cup A_1$ get reassigned to `nearest`$(\sigma^*(j))$ and so for them the difference in cost is $\leq 2d_j^*$ as in the previous two cases. Consider now a client $j \in A_t$ for $t \geq 2$. We assign such a client to i_1^*, and use triangle inequality, and the fact that i_1^* was closest to i, to bound the distance as follows.

$$\begin{aligned} d(i_1^*, j) &\leq d(i,j) + d(i, i_1^*) \leq d(i,j) + d(i, i_t^*) \\ &\leq d_j + d(i,j) + d(i_t^*, j) = 2d_j + d_j^* \end{aligned}$$

Since swapping i and i_1^* doesn't help we get

$$f_i + \sum_{j \in \Gamma(i)} d_j \leq f_{i_1^*} + \sum_{j \in A_0 \cup A_1} \left(d_j + 2d_j^*\right) + \sum_{t=2}^{k} \sum_{j \in A_t} \left(2d_j + d_j^*\right) \tag{4}$$

Note that we would have liked $2d_j^* + d_j$ for the $t \geq 2$ summands as well, but things seem swapped. Therefore, we need one extra piece of argument here. For $t \geq 2$, consider *opening* the facility i_t^* and assigning the clients in A_t to i_t^*. See the right figure in Figure 4 for an illustration. Since this doesn't help, we get

$$\forall 2 \leq t \leq k, \quad 0 \leq f_{i_t^*} + \sum_{j \in A_t} \left(d_j^* - d_j\right) \tag{5}$$

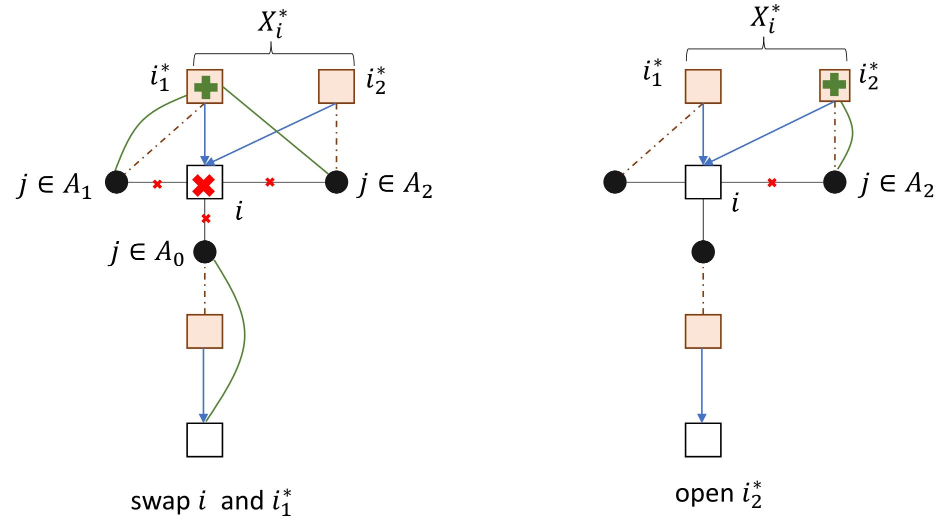

Figure 4: *Salmon squares denote facilities in X^* while empty squares denote facilities in X. Dotted brown lines denote the assignment σ^*. The blue arrows denote the* `nearest` *map from X^* to X. Green lines denote reassignments. $X_i^* = \{i_1^*, i_2^*\}$ and i_1^* is closer to i. In the left figure, we swap i and i_1^*. The client $j \in A_0$ go to* $\texttt{nearest}(\sigma^*(j))$*, the client $j \in A_1$ go to i_1^*, while the client $j \in A_2$ also goes to i_1^*. In the right figure i_2^* is opened and the client $j \in A_2$ is reassinged to it.*

And now, if we add (4) and (5), we get

$$f_i \leq \sum_{t=1}^{k} f_{i_t^*} \quad + \sum_{j \in \Gamma(i)} 2d_j^*$$

proving the lemma in this case as well.

Notes

The local search algorithm described above is from the paper [1] by Arya, Garg, Khandekar, Meyerson, Munagala, and Pandit. The analysis here is inspired by the simpler analysis in [3] by Gupta and Tangwongsan. For UFL, a slightly different local search was studied in [2] by Charikar and Guha, with the same approximation factor. As in the case of greedy algorithm, the above analysis shows that local search gives an $(2, 3)$-approximation. One can thus get a better factor by scaling and greedy augmentation tricks present in [2]. The current best approximation factor for UFL is 1.488 and is present in the paper [4] 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 [5].

References

[1] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing (SICOMP)*, 33(3):544–562, 2004.

[2] 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.

[3] A. Gupta and K. Tangwongsan. Simpler analyses of local search algorithms for facility location. *arXiv preprint arXiv:0809.2554*, 2008.

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

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