Primal-Dual Algorithm for Facility Location[1]

- ***The Facility Location Problem.*** In the (uncapacitated) facility location (UFL) problem, we are given a set F of facilities, a set C of clients, and connection costs $d(i,j)$ for $i \in F$ and $j \in C$ which measures the "distance" between i and j. 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 the nearest open facility, to minimize

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

We assume that the distances form a metric, that is, satisfy triangle inequality

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

then the problem is called the metric UFL. In this lecture, we will see an influential primal-dual 3-approximation algorithm. We begin by writing these two LPs.

- ***The Primal LP Relaxation and its Dual.***

$$\min \sum_{i \in F} f_i y_i + \sum_{i \in F, j \in C} d(i,j) x_{ij} \qquad \text{(UFL-P)}$$
$$\sum_{i \in F} x_{ij} \ge 1, \quad \forall j \in C \qquad (1)$$
$$y_i - x_{ij} \ge 0, \quad \forall i \in F, \forall j \in C \qquad (2)$$
$$x_{ij}, y_i \ge 0, \quad \forall i \in F, \forall j \in C \qquad (3)$$

$$\max \sum_{j \in C} \alpha_j \qquad \text{(UFL-D)}$$
$$\sum_{j \in C} \beta_{ij} \le f_i, \quad \forall i \in F \qquad (4)$$
$$\alpha_j - \beta_{ij} \le d(i,j), \quad \forall i \in F, \forall j \in C \qquad (5)$$
$$\alpha_j, \beta_{ij} \ge 0, \quad \forall i \in F, \forall j \in C \qquad (6)$$

The primal has two kinds of variables: y_i indicating if i is open and x_{ij} indicating if client j is connected to facility i. (1) captures the fact that every client must be connected to some facility. (2) captures the fact that no client can connect to a facility unless the latter is opened.

For each constraint of the primal, there is a dual variable. There are variables α_j's for each $j \in C$, corresponding to primal constraint (1). There are variables β_{ij} for all facility-client pairs, corresponding to primal constraint (2). One should think of α_j as a "charge" on the client j, and the β_{ij}'s as a "charge bump" on facility-client pair (i,j).

The dual objective is to maximize the total charge, that is, $\sum_{j \in C} \alpha_j$. There are two kinds of constraints corresponding to the two kinds of variables, x_{ij} and y_i, in the primal. Constraint (5) bounds the charge α_j on any client j; for any facility i, α_j is at most the distance to $d(i,j)$ to the facility *plus* the charge bump β_{ij} for this pair (i,j). Thus the charge bumps can be used to raise α_j's. However, the constraint (4) puts an upper bound on the charge bumps: for any facility i, the total charge bumps it participates in, that is $\sum_{j \in C} \beta_{ij}$ is at most the facility opening cost f_i.

[1] *Lecture notes by Deeparnab Chakrabarty. Last modified : 4th Mar, 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!

- ***The Primal Dual Schema for UFL.*** To recall the Primal-Dual schema, we wish to design an algorithm which solves the UFL problem along with finding (and, in fact, guided by) a feasible solution (α, β) to (UFL-D). We analyze the performance of the UFL solution by comparing it with the dual solution we constructed. Intuitively, we would like (a) to open facility i if and only if $f_i = \sum_j \beta_{ij}$, and (b) connect client j to facility i iff $\alpha_j = d(i,j) + \beta_{ij}$. Furthermore, we would also like (c) $\beta_{ij} > 0$ to imply j connects to i. It is not too hard to see if we achieved all three of these properties, we would in fact have an exact solution, and this is too much to hope for since UFL is an NP-hard problem. What we show below is how to obtain (a), (c), but condition (b) for only a subset of the clients; for the remaining clients, we argue about their connection costs using triangle inequality.

  Initially, all α_j's and β_{ij}'s are 0. We maintain a set of *active* clients A, which is initialized to the set of all clients C. We raise the α_j's for active clients at a uniform rate. At some point of time, we have $\alpha_j = d(i,j)$ for some facility-client pair (i,j). At this point, we raise the "charge bump" β_{ij} for this pair also at the same rate. We call a pair (i,j) *tight* if $\alpha_j = d(i,j) + \beta_{ij}$, and we say this client j "feeds" the facility i. We also maintain a set X of (tentatively) open facilities initialized to $\emptyset$. A facility i enters X iff $\sum_j \beta_{ij} = f_i$. Once a facility enters X, all clients that were "feeding" this facility are deemed inactive, and are tentatively connected to this facility i via a map σ. The above is done till all clients are inactive and thus assigned to some tentatively open facility. This completes the *first stage* of the algorithm.

  procedure UFL PRIMAL-DUAL STAGE I(UFL instance $\mathcal{I}$):
  $\quad$ $\alpha_j \leftarrow 0$ for all $j \in C$; $\beta_{ij} \leftarrow 0$ for all $i \in F, j \in C$.
  $\quad$ $A \leftarrow C$; $X \leftarrow \emptyset$; $\Pi \leftarrow \emptyset$ ▷ *Π will denote the set of tight pairs.*
  $\quad$ **while** $A \neq \emptyset$ **do**:
  $\quad\quad$ For each $j \in A$ and each $(i,j) \in \Pi$ increase α_j and β_{ij}, respectively, at a uniform rate till one of the following occurs.
  $\quad\quad$ a. A new pair (i,j) becomes tight, that is $\alpha_j = d(i,j)$. In that case,
  $\quad\quad\quad$ – $\Pi \leftarrow \Pi + (i,j)$.
  $\quad\quad\quad$ – If $i \in X$, $\sigma(j) \leftarrow i$ and $A \leftarrow A \setminus \{j\}$
  $\quad\quad\quad$ ▷ *If i was already open, the client j is assigned to this facility. If not, β_{ij} will grow for this pair.*
  $\quad\quad$ b. For some $i \in F \setminus X$, $\sum_{j \in C} \beta_{ij} = f_i$. In that case,
  $\quad\quad\quad$ – For all $j \in A$ with $(i,j) \in \Pi$, $A \leftarrow A \setminus \{j\}$ and $\sigma(j) \leftarrow i$.
  $\quad\quad\quad$ ▷ *A new facility is open, and all active clients feeding it are assigned to it.*
  $\quad$ **return** (X, σ).

- A few invariants about the above algorithm are to be noted.
  - Whenever a client j leaves A, it is assigned to $\sigma(j) \in X$. Furthermore, $(\sigma(j), j) \in \Pi$ implying $d(\sigma(j), j) = \alpha_j - \beta_{ij}$.
  - Whenever a facility i is added to X, we have $\sum_{j \in C} \beta_{ij} = f_i$.

It is instructive to *try* and analyze the cost of the above algorithm and compare it to the total dual obtained. Note that the cost of the solution is

$$\texttt{cost}(X) = \sum_{i \in X} f_i + \sum_{j \in C} d(\sigma(j), j) = \sum_{i \in X} \sum_{j \in C} \beta_{ij} + \sum_{j \in C} \left(\alpha_j - \beta_{\sigma(j),j}\right)$$

The second summation contains $\sum_{j \in C} \alpha_j$ which is what we would like to compare with. Indeed, we could rewrite the RHS as

$$\texttt{cost}(X) = \sum_{j \in C} \alpha_j + \sum_{i \in X} \left(\sum_{j \in C} \beta_{ij} - \sum_{j \in C : \sigma(j) = i} \beta_{ij} \right)$$

The second parenthesis is the reason we can't compare $\texttt{cost}(X)$ directly with $\sum_{j \in C} \alpha_j$. To see this, it is useful to imagine a scenario where this second parenthesis vanishes: this occurs if $\beta_{ij} > 0$ implies $\sigma(j) = i$. Unfortunately, this condition is *not* satisfied by the Stage I algorithm; there could be *many* clients $j \in C$ with $\beta_{ij} > 0$ who are not *not* be assigned to i. This can occur if some other $i' \in X$ was opened before i, and j was assigned to that facility. In turn, the second parenthesis term may become much larger than the $\sum_{j \in C} \alpha_j$'s.

Exercise: *Indeed, come up with an example where this actually occurs.*

To allay this, the *second stage* of the algorithm makes sure that an open facility is paid for *only* by clients which are assigned to it. And this brings us to the second idea in this primal-dual algorithm.

- ***Facility Graph and Independent Sets.*** Recall, X is the set of facilities tentatively opened by the above algorithm, and each client $j \in C$ is part of at least one tight pair (i, j) with $i \in X$. In particular, $i = \sigma(j)$ is one such facility, but there can be multiple such i's. Construct a graph $G(X, E)$ over the tentatively open facilities, where there is an edge $(i, i') \in E$ iff there is a client j with $\beta_{ij} > 0$ and $\beta_{i'j} > 0$, that is, the same client "fed" both facilities. Let I be any *maximal* independent set in the graph. The final algorithm is : open I.

procedure UFL PRIMAL-DUAL STAGE II(X, α, β from Stage I):
  Construct graph $G = (X, E)$ where $(i, i') \in E$ if and only if there exists $j \in C$ with $\beta_{ij} > 0$ and $\beta_{i'j} > 0$.
  Final any maximal independent set I in G.
  return I as the final set of facilities opened.

- *Analysis.* Every client j would connect to their closest facility in I. We prove an *upper bound* to this connection cost by describing a potentially sub-optimal assignment $\sigma : C \to I$ as follows.

  For each facility $i \in X$, let $\Gamma(i) := \{j : (i, j) \in \Pi\}$ be the tight pairs incident on i. For any set $T \subseteq X$, $\Gamma(T) = \bigcup_{i \in T} \Gamma(i)$. So, $\Gamma(T)$ is the set of clients which can be connected to some facility in T via a tight pair. Now, if a client j can be connected so to a facility in I, that is if $j \in \Gamma(I)$, then set $\sigma(j) \leftarrow i$ where $i \in I$ is such that (i, j) is tight and $\beta_{ij} > 0$. Note that by definition of the independent set, there can be *at most* one such $i \in I$. If $\beta_{ij} = 0$ for all $i \in I$, then arbitrarily pick any i such that $(i, j) \in \Pi$.

For clients $j \in C \setminus \Gamma(I)$, consider $i \in X \setminus I$ to be the facility with $(i, j) \in \Pi$ to which j was tentatively assigned to in Stage I. Since i is not in I, there must exist an edge (i', i) in $G(X, E)$. That is, there must exist a client $j' \in \Gamma(i')$ such that $\beta_{ij'} > 0$ and $\beta_{i'j'} > 0$. Assign the client j to $i' \in I$, that is, $\sigma(j) \leftarrow i'$. This completes the description of σ.

We prove the following theorem which, in particular, proves that the algorithm is a 3-approximation algorithm.

Theorem 1. $\texttt{cost}(I) \;\leq\; \sum_{j \in C} d(j, \sigma(j)) + 3\sum_{i \in I} f_i \;\leq\; 3\sum_{j \in C} \alpha_j$

Proof. As before, we can argue about the facility opening costs as

$$\sum_{i \in I} f_i = \sum_{i \in I} \sum_{j \in \Gamma(i)} \beta_{ij} \underbrace{=}_{\text{exchanging summations}} \sum_{j \in \Gamma(I)} \sum_{i \in I : \beta_{ij} > 0} \beta_{ij}$$

Now, as mentioned above, since I is an independent set, for all $j \in \Gamma(I)$ we can have $\beta_{ij} > 0$ for at most one $i \in I$, and if so, that i is precisely $\sigma(j)$. Therefore, we can simplify the RHS of the above equation to get

$$\sum_{i \in I} f_i = \sum_{j \in \Gamma(I)} \beta_{\sigma(j),j} \underbrace{=}_{(\sigma(j),j)\ \text{is tight}} \sum_{j \in \Gamma(I)} \left(\alpha_j - d(\sigma(j), j)\right) \tag{7}$$

We can now argue about the connection cost of the, possibly sub-optimal, assignment σ.

$$\sum_{j \in C} d(j, \sigma(j)) = \sum_{j \in \Gamma(I)} d(j, \sigma(j)) + \sum_{j \in C \setminus \Gamma(I)} d(j, \sigma(j)) \tag{8}$$

Note that the first term in the RHS of (8) precisely gets canceled by the negative term in the RHS of (7). Putting them together, we get

$$\sum_{i \in I} f_i + \sum_{j \in C} d(j, \sigma(j)) = \sum_{j \in \Gamma(I)} \alpha_j + \sum_{j \in C \setminus \Gamma(I)} d(j, \sigma(j)) \tag{9}$$

Now we prove the following lemma which uses triangle inequality.

Lemma 1. For every $j \in C \setminus \Gamma(I)$, $d(j, \sigma(j)) \leq 3\alpha_j$.

The proof of the theorem now follows easily from (7), (8), and Lemma 1. Indeed, one gets something potentially stronger by multiplying (7) by 3, adding it to (8), and then applying Lemma 1.

$$3\sum_{i \in I} f_i + \sum_{j \in C} d(j, \sigma(j)) \leq \underbrace{3\sum_{j \in \Gamma(I)} \alpha_j + 3\sum_{j \in C \setminus \Gamma(I)} \alpha_j}_{=3\sum_{j \in C} \alpha_j} - 2\sum_{j \in \Gamma(I)} d(j, \sigma(j))$$

□

- *Proof of Lemma 1.* Fix a client $j \in C \setminus \Gamma(I)$, and let's recall what $\sigma(j)$ is. We pick $i \in X \setminus I$ to be the facility with $(i, j) \in \Pi$ to which j was tentatively assigned to in Stage I. Since i is not in I, there must exist facility $i' \in I$ and $j' \in \Gamma(i')$ such that $\beta_{ij'} > 0$ and $\beta_{i'j'} > 0$. We then assign $\sigma(j)$ to be $i' \in I$.

  We now use triangle inequality to assert

  $$d(\sigma(j), j) = d(i', j) \leq d(i, j) + d(i, j') + d(j', i') \leq \alpha_j + 2\alpha_{j'} \tag{10}$$

  where we have used the fact that (i, j), (i, j') and (i', j') are in Π to get the second inequality.

  Now, what do we know about α_j and $\alpha_{j'}$? α_j stopped growing when j was deemed inactive. This occurs in Line 5(a) or (b), but note that at that time j is tentatively assigned to i. This must occur *after or at the same time* i was declared tentatively open (ie, added to X); if Line 5(a) it's the former, if Line 5(b) it's the latter. Let's call this "opening time" of i to be t_i, and we have argued $\alpha_j \geq t_i$.

  Now, since $\beta_{ij'} > 0$, (i, j') was already tight at the time i was declared tentatively open, that is t_i. Therefore, j' must be deemed inactive *at least* by this time; if it's still active at time t_i then it will be deemed inactive at that iteration. This implies $\alpha_{j'} \leq t_i$, and therefore, $\alpha_{j'} \leq \alpha_j$. Substituting in (10), we obtain the proof of the lemma.

Notes

The algorithm described here is from the paper [2] by Jain and Vazirani. The fact that the algorithm's cost is at most 3lp even when the facility opening costs are multiplied by factor 3 is extremely useful in designing algorithms for the k-median problem. In the same paper [2], Jain and Vazirani showed how such algorithms can be used to design approximation algorithms for the k-median problem with a hit of factor 2. This style has been further refined in the paper [3] and [1] to get the best-known approximations for the k-median problem.

References

[1] J. Byrka, T. Pensyl, B. Rybicki, A. Srinivasan, and K. Trinh. An improved approximation for k-median, and positive correlation in budgeted optimization. In *Proc., ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 737–756, 2014.

[2] K. Jain and V. V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangean relaxation. *Journal of the ACM*, 48(2):274–296, 2001.

[3] S. Li and O. Svensson. Approximating k-median via pseudo-approximation. *SIAM Journal on Computing (SICOMP)*, 45(2):530–547, 2016.