On primal-dual and dual-fitting of the FTFP problem

UCR theory lab

December 30, 2012

1 H_n-approximation of Star-greedy on FTFP

In this section we show that the star-greedy algorithm which repeatedly picking the best star (the one with minimum average cost) gives an approximation ratio of $H_n = \ln(n)$ where $n = |\mathcal{C}|$ is the number of clients.

When we run the star-greedy algorithm, for every client j, we associate each demand of j with a number α_j^l, which is the average cost of the star when l^{th} demand of j is connected. Now we let $\alpha_j = \alpha_j^{r_j}$, that is, take α_j to be the finishing α_j^l, and order clients by increasing α_j. That is,

$$\alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_n$$

Due to the algorithm, for every $j = 1, \ldots, n$, we have

$$\sum_{l=j}^{n} (\alpha_j - d_{il})_+ \leq f_i$$

for every facility i. The reason is that, when the last demand of j is connected, all clients $j+1, \ldots, n$ are still active so their total contribution cannot exceed f_i.

Now we take a closer look at the numbers $\{\alpha_j\}$. We know that the algorithm's total cost is exactly $\sum_{j=1}^n \sum_{l=1}^{r_j} \alpha_j^l$, which is no more than $\sum_{j=1}^n r_j \alpha_j$ since we take α_j to be $\alpha_j^{r_j}$. Now if we can show that $\sum_{j=1}^n r_j \alpha_j$ is no more than $\gamma \cdot \text{OPT}$, where OPT is the cost of an integral optimal solution to the given FTFP instance, then we claim our algorithm returns an integral solution within a factor of γ.

We show that $\sum_{j=1}^n r_j \alpha_j$ is within a factor of γ from OPT by showing that $\{\alpha_j / \gamma\}$ is a feasible dual solution to the following program, which is the dual program of the primal LP for FTFP.

$$\begin{aligned} \max \ & \sum_j r_j \alpha_j \\ \text{subject to: } & \sum_{j=1}^{n} (\alpha_j - d_{ij})_+ \leq f_i \text{ for every facility i} \end{aligned}$$

To find the minimum γ that would shrink $\{\alpha_j\}$ to a feasible dual solution, we need to find a worst case instance to maximize γ, also it is clear that the worst case instance must contain a star whose feasibility requirement would achieves the value of γ, and this star would be the worst star in that instance.

As a first step we can drop the $\max\{0, \cdot\}$, because we can always find a new star by dropping those j with $\alpha_j - d_{ij}$ term negative, and that new star would still be a worst case star. Suppose a worst case star has k clients, and is with facility i, then we have

$$\sum_{j=1}^{k} \alpha_j - d_{ij} \leq f_i$$

Here we rename clients in the new star to be $1, \ldots, k$, although among them, they are still ordered by their α_j.

Now our goal is to find a supremum of the following program:

$$\max \frac{\sum_{j=1}^{k} \alpha_j}{f_i + \sum_{j=1}^{k} d_{ij}}$$
$$\text{subject to: } \sum_{l=j}^{k} (\alpha_l - d_{il})_+ \leq f_i \text{ for } j = 1, \ldots, n$$

Since we are dealing with a particular star, we can abstract away i, to obtain the following program:

$$\max \frac{\sum_{j=1}^{k} \alpha_j}{f + \sum_{j=1}^{k} d_j} \tag{1}$$
$$\text{subject to: } \sum_{l=j}^{k} (\alpha_j - d_l)_+ \leq f \text{ for } j = 1, \ldots, n$$

Now we claim we can drop the $\max\{0, \cdot\}$ operator because this would relax the constraint in (1) and can only make objective value larger (since we are maximizing), so the real optimal is upper bounded by the relaxed optimal. This allows us to work on

$$\max \frac{\sum_{j=1}^{k} \alpha_j}{f + \sum_{j=1}^{k} d_j} \tag{2}$$
$$\text{subject to: } \sum_{l=j}^{k} (\alpha_j - d_l) \leq f \text{ for } j = 1, \ldots, n$$

For each $j = 1, \ldots, n$, the constraint above simply can be rewritten as

$$(k - j + 1)\alpha_j \leq f + \sum_{l=j}^{k} d_l \leq f + \sum_{l=1}^{k} d_l. \tag{3}$$

The first inequality is a rewrite of the constraint in (2) and the second is straightforward.

Therefore we have $\alpha_j \leq (1/(k-j+1))(f + \sum_{j=1}^{k} d_j)$, and it easily follows that

$$\sum_{j=1}^{n} \alpha_j \leq (1/k + 1/(k-1) + \ldots + 1) = H_k \leq H_n = \ln(n) \tag{4}$$

2 Dual-fitting Analysis on FTFP

This section gives a simple example that the dual-fitting analysis of a greedy algorithm which repeatedly picking the most cost-effective star (the star with minimum average cost) is unlikely to give the same ratio as that for the UFL problem.

The example consists of 1 facility with cost $f_1 = n$, and n clients with demands $r_1 = r_2 = \ldots = r_{n-1} = 1$ and $r_n = n$. All $d_{ij} = 0$. Now running the star-greedy algorithm, we will first pick a star with all n clients and we open 1 copy of facility f_1. We then have only client n with residual demand $r'_n = n-1$, and we have no other option but to open facility f_1 for another $n-1$ copies.

Now the dual-fitting based analysis will associate each demand with a dual variable $\alpha_j^1, \ldots, \alpha_j^{r_j}$ and the proposed dual solution is $\bar{\alpha}_j = \sum_{l=1}^{r_j} \alpha_j^l / r_j$ and try to find a minimum γ such that $\{\bar{\alpha}_j/\gamma\}$ is a feasible dual, that is

$$\sum_{j=1}^{n} (\bar{\alpha}_j/\gamma - d_{1j})_+ \leq f_1 = n \tag{5}$$

which is

$$\sum_{j=1}^{n} \bar{\alpha}_j/\gamma \leq n \tag{6}$$

since all $d_{ij} = 0$ and $f_1 = n$.

From the greedy algorithm, we have $\alpha_j^1 = 1$ for $j = 1, \ldots, n$, and $\alpha_n^l = n$ for $l = 2, \ldots, n$. Therefore $\bar{\alpha}_j = 1$ for $j = 1, \ldots, n-1$ and $\bar{\alpha}_n = (1 + (n-1)n)/n = n-1$. The shrinking factor γ we seek thus satisfies

$$\sum_{j=1}^{n-1} (\bar{\alpha}_j/\gamma - 0)_+ + (\bar{\alpha}_n/\gamma - 0)_+ \leq f_1 = n, \tag{7}$$

which is

$$(n-1)(1/\gamma) + (n/\gamma) \leq n \tag{8}$$

Simple algebra will show that γ can be made arbitrarily close to 2 when n is large. On the other hand we know that the same greedy algorithm with dual-fitting analysis gives a ratio of 1.81 for the UFL problem where all $r_j = 1$.

This example does not actually rule out the possibility to prove a constant ratio of the star-greedy algorithm on FTFP. In fact greedy gets exactly the same solution as the optimal integral solution for this example. All it says is that the dual-fitting analysis on greedy algorithm, when applied to the FTFP or FTFL problem, cannot possibly give a ratio much better than 2. And this partly explains why generalizing primal-dual or dual-fitting algorithms from UFL to fault-tolerant problems like FTFL or FTFP is not successful when r_j's are not equal, that is demands are not uniform. Intuitively there seems to be some issue fundamental to the dual-fitting approach as the proposed dual solution $\bar{\alpha}_j$'s can be very different between each other so shrinking all of them by a common factor γ might not give a strong upper bound on the approximation ratio. It is also quite possible that the example may be strengthened to show that dual-fitting cannot achieve a worse yet ratio.

3 Dual-fitting Analysis can be H_n on FTFP

This section extends the idea in Section 2 to show that dual-fitting based analysis on greedy algorithm can be off by a factor as large as $H_n = \ln(n)$. This complements the $\ln(n)$ upper bound shown in Section 1.

The example has one facility with $f_1 = 1$, all $d_{ij} = 0$. There are n clients with demands $r_1, r_2, \ldots, r_n$ with $r1 \ll r_2 \ll \ldots \ll r_n$. Now following the idea in Section 2, we shall have the proposed dual variable with value $\alpha_1 = 1/n, \alpha_2 = 1/(n-1), \alpha_3 = 1/(n-2), \ldots, \alpha_n = 1$. We take the α_j value to be the average of the cost of individual demand of a client, which is α_j^l, so that the algorithm's cost is equal to $\sum_j r_j \alpha_j$, since the algorithm's cost is equal to $\sum_{j=1}^{n} \sum_{l=1}^{r_j} \alpha_j^l$ by definition. Now we need a number γ such that $\{\alpha_j/\gamma\}$ form a feasible dual solution, that is, we need $\sum_{l=1}^{n} \alpha_j \leq f_1 = 1$. It is easily seen that γ needs to be at least $H_n = \ln(n)$

4 Star-Greedy and Dual-fitting on FTFL

In this section we attempt to carry the claims and arguments developed in earlier sections for FTFP to apply to FTFL, a better known problem.

Recall in FTFL, we are given a set $\mathcal{F}$ facilities and a set of $\mathcal{C}$ of clients, with each client j having demand r_j, meaning it needs to be connected to r_j different facilities. Each facility i can be opened once with cost f_i. We are also given d_{ij} satisfying the triangle inequality. The main difference between FTFL and FTFP is that FTFP allows an arbitrary number of facilities to be opened on the same site i, each paying f_i. This results in an extra constraint in the FTFL LP formulation.

$$\begin{aligned}
\text{minimize } & \sum_{i\in\mathcal{F}} f_i y_i + \sum_{i\in\mathcal{F}, j\in\mathcal{C}} d_{ij}x_{ij} \\
\text{subject to: } & y_i \geq x_{ij} \\
& \sum_{i\in\mathcal{F}} x_{ij} \geq r_j \\
& y_i \leq 1
\end{aligned}$$

and the dual program is

$$\begin{aligned}
\text{maximize } & \sum_{j\in\mathcal{C}} r_j\alpha_j - \sum_{i\in\mathcal{F}} z_i \\
\text{subject to: } & \sum_{\beta_{ij}} \leq f_i + z_i \\
& \alpha_j - \beta_{ij} \leq d_{ij}
\end{aligned} \tag{9}$$

Now we use the same algorithm as before by greedily picking the star with minimum average cost until all clients have been connected to r_j different facilities. This will again give us a sequence of α_j^l numbers associated with each connection of a client j. We could still order clients by $\alpha_j^{r_j}$, that is, their last α_j value, or the average cost of the star that a client makes its last connection.

Now we set the dual solution as $\alpha_j = \alpha_j^{r_j}$ and $z_i = 0$. Then the objective of this dual solution is an upper bound on the algorithm's cost, which is $\sum_{j\in\mathcal{C}}\sum_{l=1}^{r_j}\alpha_j^l$. We then show that $(\alpha_j/H_n, z_i/H_n)$ is a feasible dual to (9), thereby establishing the approximation ratio being H_n. The proof is very similar to that in Section 1.

On the other hand, the example in Section 2 can also be used for FTFL, thus showing that greedy algorithm analyzed using dual-fitting cannot give a ratio better than 2. Recall that for UFL where all $r_j = 1$, the greedy algorithm was shown to have a ratio of 1.81 using dual-fitting analysis.

The result compares favorably to the first result on FTFL, in which Jain and Vazirani showed that FTFL can be approximated to a ratio of $3H_n$ using a generalized version of the JV algorithm. The result of H_n improves it by a factor of 3.

5 $O((\log R/\log\log R)^2)$*approximation*

It appears that FTFP can be approximated to ratio $O((\log R/\log\log R)^2)$ quite easily, by taking advantage that an instance with $R = \max_j r_j$ can be approximates to ratio no more than R easily.

The idea is to pick k such that $k^k = R$ where $R = \max_j r_j$. Now we group clients by their r_j such that $[1,k), [k,k^2), \ldots, [k^{k-1}, k^k]$ so we have k groups. Solve each group using any primal-dual algorithm gives a partial solution with at most k times opt of that partial instance. Now combine the partial solutions we obtain one integral solution. Notice that each optimal integral solution to the original instance can have each of its facility duplicated k times and then decompose into feasible integral solutions to each of the sub-instances. This shows that the combined solution has cost no more than k times overall opt. And in solving each sub-instance, we find a solution no more than k times the opt for that sub-instances. Overall we have a solution with ratio $k * k = k^2$, which is better than $\log R$.

6 Future Directions

It seems the greedy algorithm is likely to give a constant ratio, although two things have to happen:

- We need to use an analysis different from dual-fitting.
- We have to use the triangle inequality somewhere, without that the approximation ratio cannot be better than H_n and examples from non-metric UFL already shows a lower bound of H_n in approximation.

7 On $O(1)$-approximation on Greedy Algorithm

Now we seek to show a ratio of $O(1)$ for the greedy algorithm that repeatedly picks the best star until all clients are fully-connected.

The special instance we consider is the example that we use to show greedy using dual-fitting analysis can be a factor of H_n away from being dual-feasible. In this example, there is one site with $f_i = 1$ and all clients are co-located at the same point of that site, that is $d_{ij} = 0$ for all i, j. There are n clients and their demands are $r_1 \leq r_2 \leq \ldots \leq r_n$. Then the greedy algorithm will first create r_1 facilities and each of the n clients will make r_1 connections to them. The next round will create another $r_2 - r_1$ facilities and clients 2 to n will make that many connections and so on, until the n^{th} round client n will create $r_n - r_{n-1}$ new facilities and connect to them.

The algorithm only concerns which star to pick and how to make connections, it does not specify how to share cost among participating clients of a star, and that is part of the analysis. We now look for a charging scheme that would resolve the H_n shrinking factor and keep it to a constant. The scheme at a high level is to group clients with the same r_j and then distribute facility cost proportional to r_j to each group. Within a group we distribute cost to clients evenly. For the convenience of analysis, we actually round each r_j up to the nearest power of 2, that is $r'_j = 2^k$ for some k such that $2^{k-1} < r_j \leq 2^k$, and we will work on r'_j. This will at most double the total cost.

According to our scheme, w.l.o.g. we can assume r'_j are distinct and they are all powers of 2. Let $R_k = \sum_{j=k}^{n} r'_j$, then client 1 has its share being r'_1/R_1, it has r'_1 demands to satisfy and therefore its $\alpha_1 = r'_1/R_1$. Similarly for demand 2, we have $\alpha_2 = 1/r'_2(r'_1 \cdot r'_2/R_1 + (r'_2 -' r'_1) \cdot r'_2/R_2) = r'_1/R_1 + (r'_2 - r'_1)/R_2$. If we sum all α_j for $j = 1, \ldots, n$, we obtain

$$S_n = n \cdot r'_1/R_1 + (n-1)(r'_2 - r'_1)/R_2 + \ldots + (r'_n - r'_{n-1})/R_n \tag{10}$$

It is easy to see that

$$\begin{aligned} S_n &= n \cdot r'_1/R_1 + (n-1)(r'_2 - r'_1)/R_2 + \ldots + (r'_n - r'_{n-1})/R_n \\ &\leq n \cdot r'_1/R_1 + (n-1)r'_2/R_2 + \ldots + r'_n/R_n \\ &\leq n \cdot r'_1/r'_n + (n-1)r'_2/r'_n + \ldots + r'_n/r'_n \\ &\leq n/2^n + (n-1)/2^{n-1} + \ldots + 1 \\ &\leq 2 = O(1) \end{aligned}$$