

Lecture Outline:

- K -median
- Filtering algorithm for K -median
- Local search algorithm for K -median

This lecture describes the problem of K -median. A filtering algorithm with $(2, 6)$ -approximation ratio will be described briefly. And also, an approximation algorithm based on local search will be discussed, with a 5-approximation ratio generally [AGK⁺04], which is 3-approximation for some special case.

1 K -median

K -median problem is similar to the facility location problem, with the difference that instead of a cost for each facility, here is a bound k on the number of facilities. Here is the definition of k -median problem:

Problem 1. Given a set of locations(clients) V , a set \tilde{F} of facilities, a cost function between two locations $C : (V \cup \tilde{F}) * (V \cup \tilde{F}) \rightarrow Q^+$ and an input parameter k , $0 < k \leq |\tilde{F}|$, the k -median problem is to find a subset $F \subseteq \tilde{F}$ and $\sigma : V \rightarrow F$ where the following should be satisfied:

- $|F| = k$
- The total cost $\sum_{i \in V} C_{i\sigma(i)}$ is minimized.

And in the form of LP, we have:

$$\begin{aligned}
 & \min \sum_{i,j} C_{ij} x_{ij} \\
 & s.t. \quad \sum_j x_{ij} \geq 1, \quad \forall i \\
 & \quad \quad x_{ij} \leq y_{ij}, \quad \forall i, j \\
 & \quad \quad \sum_j y_i \leq k
 \end{aligned} \tag{1}$$

Compared to the facility location, the major challenge of K -median is to satisfy the constraints. This means that we should keep a lower cost while simultaneously satisfying the condition that we can have at most k facilities selected in our solution.

2 Filtering algorithm for K -median

We can use the filtering algorithm to get an approximation solution for K -median, which is similar to the one we has used in facility location problem. Still, after get the LP solution for the K -median problem, we will use randomized rounding, based on the filtering technology.

Specially, for each client i , we will maintain a ball, whose center is the client i , with the radium

$$r_i = 2 \sum_j c_{ij} x_{ij}^* \quad (2)$$

Here x^* is the optimal solution of the corresponding LP. Then we will sort all the balls in a non-increasing order. Each time for the non-overlapping ball with the maximal radium, pick a facility location in it and open it, then remove this ball and all the other balls overlapping with it. This process will be continued recursively until k facilities have been opened.

Just like the analysis in set cover problem, this algorithm can produce a $(2, 6)$ -approximation solution, which means that the solution is 2-approximation on the facilities' open cost and 6-approximation on the clients' connection cost.

3 Local search algorithm for K -median

In this section we will introduce the local search method to solve the K -median problem with the approximation ratio of 5 generally, which will have a 3-approximation ratio in some special case.

The basic idea of local search is: we start with an arbitrary set of k facilities, which is not necessarily optimal, then we will try to improve this solution by swapping the facilities selected in our initial solution with other facilities until no improvement can be made by any swapping. Then we will get a solution S , which is locally optimal. Our goal then is to understand how good a local optimization is.

We will define the "improvement" of a swap like this: If for each facility selected in both local optimal solution and the global optimal solution, we associate a neighborhood, defined as following:

$$\begin{aligned} N_S(s) &= \{i \in V, \sigma(i) = s\} \\ N_{OPT}(o) &= \{i \in V, \sigma^*(i) = o\} \end{aligned}$$

Here $\sigma^*(i)$ is the optimal location of the facility serving the client i . So when we are adding $o \in OPT$ to our local solution S , the improvement means:

$$cost(S) - cost(S + \{o\}) \geq \sum_{i \in N_{OPT}(o)} [C_{i\sigma(i)} - C_{i\sigma^*(i)}] \quad (3)$$

If we consider adding each facility of OPT individually to S , we have

$$\sum_{o \in OPT} [cost(S) - cost(S + \{o\})] \geq cost(S) - cost(OPT) \quad (4)$$

Then we need to know how to bound the impact of dropping a facility of S . Toward this end, we define the notion of **capture**:

Definition 1. For a local solution S and optimal solution OPT , we say $s \in S$ **captures** $o \in OPT$ if in S at least half of o 's clients are served by s , i.e.,

$$|N_S(s) \cap N_O(o)| > \frac{1}{2}|N_O(o)|$$

We will then consider two cases. Note that a facility $o \in OPT$ is captured by at most one facility $s \in S$.

- A facility $s \in S$ captures exactly one $o \in OPT$. In this situation, we just swap this pair (s, o) .
- Some of the facilities in S capture more than one facilities in OPT , and some of the choices in OPT may not have been captured by any facility in S . Here none of the facilities $s \in S$ which captures more than one optimal choices should be swapped, in such a way that no facility in S is swapped with more than 2 facilities in OPT , instead we will swap the facilities in the optimal solution with the facilities in S which capture no facilities in OPT .

Here we will show that the local algorithm described above is a 3-approximation algorithm, if all facilities in OPT were of CASE 1.

The challenge in our analysis in the swapping process is that after swapping, some of the clients locations which was served by some facilities in our initial solution should be reassigned to other facilities since the old ones have been replaced by ones from the optimal solution. So here we should find a way to estimate the change of the cost in this reassignment process. So for $i \in N_S(s) - N_{OPT}(o)$ in a swapping on (s, o) , we use a mapping function $\pi : V \rightarrow V$ to map $i \notin N_{OPT}(o)$ to $\pi(i) \in N_{OPT}(o)$, such that:

1. $\sigma^*(\pi(i)) = \sigma^*(i)$
2. $\sigma(\pi(i)) = \sigma(i)$ only if $\sigma(i)$ captures $\sigma^*(i)$

So for a given swapping between (s, o) , since S is locally optimal, we have

$$cost(S) - cost(S + \{o\} - \{s\}) \leq 0 \quad (5)$$

Since we need to add o while dropping s , for the clients who are served only by s but not o , the cost will increase since they should be reassigned. So

$$\begin{aligned}
cost(S) - cost(S + \{o\} - \{s\}) &= \sum_{i \in N_{OPT}(o)} (C_{i\sigma(i)} - C_{i\sigma^*(i)}) \\
&+ \sum_{i \in N_S(s), i \notin N_{OPT}(o)} (C_{i\sigma(i)} - C_{i\sigma^*(i)} - C_{\pi(i)\sigma(\pi(i))} - C_{\pi(i)\sigma^*(\pi(i))})
\end{aligned} \tag{6}$$

Then for the clients $i \in N_S(s) - N_{OPT}(o)$, after the swapping of (s, o) , i should be reassigned. Here we will map i to $j = \pi(i) \in N_{OPT}(o)$. According to triangle inequality, the cost of reassigning can be estimated

$$\begin{aligned}
C_{i\sigma^*(i)} + C_{j\sigma^*(j)} + C_{j\sigma(j)} - C_{i\sigma(i)} &\leq C_{i\sigma^*(i)} + C_{j\sigma^*(j)} + C_{i\sigma^*(i)} + C_{j\sigma^*(j)} + C_{i\sigma(i)} - C_{i\sigma(i)} \\
&= 2(C_{i\sigma^*(i)} + C_{j\sigma^*(j)})
\end{aligned} \tag{7}$$

From Equation 6 and Equation 7, we can get

$$\sum_{i \in N_O(o)} (C_{i\sigma(i)} - C_{i\sigma^*(i)}) \leq \sum_{i \in N_S(s), i \notin N_O(o)} 2(C_{i\sigma^*(i)} + C_{j\sigma^*(j)}) \tag{8}$$

For the situation that all facilities in OPT were of CASE 1, according to the swap policy mentioned above, all the facility $s \in S$ and also $o \in O$ will be swapped exactly once, since for each $o \in O$, there is exactly one $s \in S$ which captures it.

And also, since in the process of swapping for each pair (s, o) the clients $i \in N_{OPT}(o)$ will be at least half of the clients in $s \in S$, there will be at most half of the clients $i \in N_S(s) - N_{OPT}(o)$ for all the pairs swapped. After finishing all the swaps, there will totally be at most half of the clients i which should be considered separately. And also, $j = \pi(i)$ is a one-to-one mapping, so for the right hand side of Equation 8, we have

$$\sum_{i \in N_S(s), i \notin N_O(o)} 2(C_{i\sigma^*(i)} + C_{j\sigma^*(j)}) \leq 2 * cost(OPT) \tag{9}$$

From Equation 5, 8 and 9, we will get

$$cost(S) \leq 3 * cost(OPT)$$

For CASE 2, the policy is feasible since if we let the number of optimal choices in CASE 2 are l , there are at least $l/2$ facilities in S which will not capture any of the facilities in S , because in this case the facilities in S which capture some optimal choices will capture at least 2 facilities in OPT . We then have a set of k swaps such that each facility in OPT is swapped in once and each facility in S is swapped out once. Now we will show that the algorithm will produce a 5-approximation solution in this case.

The policy for CASE 2 will guarantee that all the $s \in S$ will be swapped at most twice, since for the facilities in OPT at least of two of which are captured by one facility in S . So for the right hand side of Equation 8, the total number of clients $i \in N_S(s) - N_{OPT}(o)$ for all the pairs will be at most the number of all the clients. So we have

$$\sum_{i \in N_S(s), i \notin N_O(o)} 2(C_{i\sigma^*(i)} + C_{j\sigma^*(j)}) \leq 4 * cost(OPT) \quad (10)$$

Then we combine Equation 5 with 10, we will get

$$\begin{aligned} 4 * cost(OPT) &\geq cost(S) - cost(OPT) \\ cost(S) &\leq 5 * cost(OPT) \end{aligned} \quad (11)$$

The in order to keep our algorithm running into the polynomial capability, we should stop the iteration process at some point when the improvement from the swap is too trivial for the cost of running time. Assume that after polynomial running time, the improvement is bounded at $\varepsilon cost(S)$, then

$$cost(S) - 5 * cost(OPT) \leq \varepsilon cost(S) \quad (12)$$

So we can get the approximation of this local search methods

$$cost(S) \leq \frac{5 * cost(OPT)}{1 - \varepsilon} \approx (5 + 5\varepsilon) * cost(OPT) \quad (13)$$

Now we will show a stop criteria so that our algorithm can stop in polynomial time, with the following claim:

Claim 1. *The local searching will stop if no swap improves cost by more than $\frac{\varepsilon * cost(S)}{k}$.*

From this criteria we have

$$cost(S') \geq (1 - \frac{\varepsilon * cost(S)}{k}) \quad (14)$$

Here $\frac{\varepsilon * cost(S)}{k}$ is the drop-in cost in each iteration of local searching. k is a polynomial in $|\tilde{F}|$ and $|V|$. According to [AGK⁺04], this will keep the pace that after the first iteration, the local searching will only check at most a polynomial rate of the previous time. Then if we have the initial local searching solution as S_{INIT} , then after t iterations,

$$cost(S_{INIT}) * \frac{\varepsilon * cost(S)}{k}^t \geq cost(OPT)$$

With the approximation that

$$\log \frac{1}{1 - \frac{\varepsilon}{k}} \approx \frac{k}{\varepsilon}$$

We will have the estimate on t

$$t \approx \frac{k}{\varepsilon} \log \frac{\text{cost}(S_{INIT})}{\text{cost}(OPT)} \approx O\left(\frac{k}{\varepsilon} \log n\right)$$

References

- [AGK⁺04] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004.