# K-Center Problems

Student Name: Yajur Ahuja
Roll Number: 2016121

BTP report submitted in partial fulfillment of the requirements
for the Degree of B.Tech. in Computer Science & Engineering
on $3^{rd} June, 2020$

**BTP Track**: Research

**BTP Advisor**
Dr Syamantak Das

Indraprastha Institute of Information Technology
New Delhi

## Student's Declaration

I hereby declare that the work presented in the report entitled **K-Center Problems** submitted by me for the partial fulfillment of the requirements for the degree of *Bachelor of Technology* in *Computer Science & Engineering* at Indraprastha Institute of Information Technology, Delhi, is an authentic record of my work carried out under guidance of **Dr. Syamantak Das**. Due acknowledgements have been given in the report to all material used. This work has not been submitted anywhere else for the reward of any other degree.

**..............................** **Place & Date:**

**Yajur Ahuja**

## Certificate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

**..............................** **Place & Date: ..............................**

**Dr. Syamantak Das**

**Abstract**

We consider a generalization of the classical K-center problem with capacities, demands, and outliers. The capacitated K-center problem adds a constraint to the original K-center problem of every vertex having a capacity assigned to it. This means that if a vertex is selected as a center, then it can serve as a center to at most its capacity. The K-center with demand version where each vertex $i$ is required to have a set of $\alpha_i$ ($\alpha_i <= k$) centers close to it. We define the Non-uniform Demand version where values of $\alpha_i$ vary for each vertex. We try to solve the capacitated $K$-center with non-uniform demand problem which is a generalization of the capacitated and the non-uniform demand versions of the $K$-center problems stated above. We also try to understand and solve another generalization of allowing outliers in the $K$-center problem.

Keywords: Algorithms, Approximation Algorithms, Integer Programming, Linear Programing, Relaxation & Rounding

## Acknowledgments

I am grateful to Dr. Syamantak Das for giving me an opportunity of working with him and introducing me to the problem. I would also like to thank Rahul Gangopadhyay for being an integral part of the project. I would like to thank them for constantly guiding me through the problem and sharing ideas for the research.

# Contents


**1 Introduction** **iv**

**2 Preliminaries** **v**

2.1 Problem . . . . . v

2.2 Reduction to an unweighted problem using LP relaxation . . . . . v

2.3 Integrality Gap . . . . . v

**3 Uncapacitated K-Center with Non-Uniform Demand** **vi**

3.1 Problem . . . . . vi

3.2 LP formation . . . . . vi

3.3 Algorithm . . . . . vii

3.3.1 Idea . . . . . vii

3.3.2 Clustering . . . . . vii

**4 Uncapacitated K-Center with Outliers** **ix**

4.1 Problem . . . . . ix

4.2 LP formation and relaxation . . . . . ix

4.3 Algorithm . . . . . x

**5 Capacitated K-Center with Non-Uniform Demand** **xv**

5.1 Problem . . . . . xv

5.2 Preliminary Definitions . . . . . xv

5.3 LP formation and relaxation . . . . . xv

5.3.1 Program formation . . . . . xv

5.4 Algorithm for Rounding . . . . . xvi

5.4.1 Clustering . . . . . xvi

5.4.2 Greedy Assignment . . . . . xvii

5.5 The Flow Network . . . . . xviii

5.6 Constant Approximation . . . . . xix

**6 Uncapacitated K-center with non-uniform demand and outliers** **xxi**

6.1 Problem . . . xxi

6.2 LP formation and relaxation . . . xxi

6.3 Iterative rounding . . . xxii

6.4 Integrality gap . . . xxiii

**7 Future Work** **xxv**

# Chapter 1

# Introduction

The k-center problem is a classic facility location problem and is defined as follows: given a finite set of vertices $V$ and a symmetric distance (cost) function $d : V \times V \rightarrow R_{\geq 0}$ satisfying the triangle inequality, find a subset $S \subseteq V$ of size $k$ such that each vertex in $V$ is close to some vertex in $S$. More formally, once we choose $S$ the objective function to be minimized is $\max_{v \in V} \min_{u \in S} d(v, u)$. The vertices of S are called centers. There are a lot of generalizations to the classic k-center problem by adding constraints like vertex capacities, demand or fault-tolerance or relaxing constraints by allowing outliers. The capacitated setting, we give a capacity function $L : V \rightarrow \mathbb{Z}_{\geq 0}$ and at most $L(u)$ vertices (clients) can be assigned to a chosen center at $u \in V$. In demand/fault-tolerant version of the K-center, we give a demand function $D : V \rightarrow \mathbb{Z}_{\geq 0}$ and each vertex $v \in V$, is required to have a set of $D(v)(D(v) <= k)$ centers close to it. Khullar and Sussman have provided algorithms on two of the variations, the Fault-tolerant K-Center Problems [4] giving an approximation of 3 and the Capacitated K-center Problem [5] giving an approximation of 5 and 6. We have found 3 approximation LP based solution for the Uncapacitated K-center with Non-Uniform Demand. Cygen et al. [3] uses Linear Programming to find an approximation algorithm. They find a way to make the LP integrality gap a constant and provide proof for it. They also provide with a way uses a caterpillar structure to perform the aggregation and transfers so the aggregation is bounded by a constant. An et al. [1] have solved the problem for the Capacitated Case using Linear Programming and a tree structure which is a generalization of the caterpillar structure and reach a 9-approximation algorithm. We extend their idea of tree structure to solve the Capacitated K-center with non-uniform Demand and have been able to obtain a constant factor approximation.

We also introduce the idea of outliers. Consider a classical K-center problem, but instead of serving all the clients, given an integer $p(\leq n)$, we will find the min distance required by k centers to to select and serve exactly p clients. The clients that disregarded are called outliers. Chirakar et al. [2] found a 3-approximation greedy solution for the K-center with outliers. We have found a 3-appoximation alogrithm using LP rounding which is client centric. In the end, we give an example showing that there exists an integrality gap between the LP solution and the optimal solution for K-center with outliers and uniform demand.

# Chapter 2

# Preliminaries

## 2.1 Problem

Consider completer graph with n vertices. We embed this graph in $\mathbb{R}^2$. Each vertex is mapped to a point in $\mathbb{R}^2$. Let $u, v$ be two vertices mapped to two distinct points. The length of the edge $(u, v)$ is the Euclidean distance between the two corresponding points. We define the distance between two vertices as the edge cost $c : V \times V \to R_{\geq 0}$. We have to choose and open K centers and assigning each vertex to an open center which minimizes the maximum distance between a vertex and its centers.

## 2.2 Reduction to an unweighted problem using LP relaxation

We will use the method used in [1]. We need to determine the lower bound $\tau^*$ on the optimal solution. We make a guess that OPT is $\tau$ adn determine if $\tau$ <OPT. We convert the graph into an undirected graph $G_{\leq\tau}(V, E)$ where we only connect those edges whose lenght is less than or equal to $\tau$. The edge cost becomes $E_{\leq\tau} =: \{(u, v)|c(u, v) \leq \tau\}$. We guess $\tau$ and try to solve the LP for the graph $G_{\leq\tau}$ which is different for each type of problem and obtain the $\tau^*$ which is the lowest value of $\tau$ for the LP solution that exists. We can say that the integral optimal solution is at least $\tau^*$. Once we have the $\tau^*$, we know which undirected graph to work with. We apply the algorithms on the undirected graph obtained.

## 2.3 Integrality Gap

There may be an integrality gap $\frac{OPT}{\tau}$ for the graph. We use the approach of Cygan et al. [3]. We consider connected components $G_{\leq\tau}$ and solve and find the minimum $k_i$ for each connected components $G_i$ separately. If $\sum k_i > k$, we have to take a larger $\tau$. We find the smallest $\tau$ for which $\sum k_i \leq k$ and the solution exists($\tau^*$).

# Chapter 3

# Uncapacitated K-Center with Non-Uniform Demand

## 3.1 Problem

Consider an undirected graph $G = (V, E)$ with n vertices. Given an integer $k$ with a demand function defined on $V$, $D : V \to \mathbb{N}$, k-center problem with non-uniform demand is to choose $k$ vertices to open, along with an assignment of every vertex $v$ to $D(v)$ different open centers such the maximum distance between a vertex and its centers is minimized.

## 3.2 LP formation

We consider an integer program to represent our non-uniform demand uncapacitated version of the K-center problem on the undirected graph $G = (V, E)$. We use the following variables in our IP

- Opening Variable($y_u$): an indicator variable for opening a center at $u$ where $u \in V$.
- Assignment Variable($x_{uv}$): an indicator variable denoting that $v$ is assigned to center $u$ where $u, v \in V$.
- Demand Function($D(v)$): defined above.

$$\sum_{u \in V} y_u \leq k; \tag{3.1}$$

$$x_{uv} \leq y_u \;\; \forall u, v \in V; \tag{3.2}$$

$$\sum_{u:(u,v) \in E} x_{uv} \geq D(v) \;\; \forall v \in V; \tag{3.3}$$

$$x, y \in \{0, 1\} \tag{3.4}$$

We will consider the LP relaxation of the IP by relaxing the variables $x, y$ to

$$0 \leq x, y \leq 1 \tag{3.5}$$

## 3.3 Algorithm

In this section we describe an algorithm which gives an approximation factor of 3 for the non-uniform demand uncapacitated K-center. We guess and find out $\tau^*$ using the method described in the previous chapter.

### 3.3.1 Idea

Our algorithm involves introducing the notion of balls and clusters which we formally define below.

**Definition 3.3.1** *We define the Ball Function $B_r : V \to 2^V$, where $B_r(v)$ is a set of all vertices that lie on or inside the ball(circle) of radius $r$ drawn around the vertex $v$. All the vertices belonging to $B_r(v)$ will be at a distance of atmost of $r$ from $v$.*

**Definition 3.3.2** *We define the Cluster of a vertex $\overline{v}$ as $C_{\overline{v}} = \{v | B_r(\overline{v}) \cap B_r(v) \neq \phi\}$*

We draw a ball of radius $\tau^*$ around each vertex in the graph. As the LP gives us a solution of distance $\tau^*$, we know that each vertex $v$ is only served by the vertices which belong to $B_{\tau^*}(v)$. We cluster the neighbouring balls greedily and for each cluster we open the number of centers required to serve the vertex with the highest demand. We then reassign the rest of the vertices to these opened centers.

### 3.3.2 Clustering

**Algorithm 1 Greedy Clustering based on Demands**

```
0: procedure CLUSTER(G)
     C = {}
     V̄ = {}
     V ← Set of all vertices in G
     while V ≠ φ do
       v̄ ← arg max_{v∈V} D(v)
       V̄ = V̄ ∪ {v̄}
       C ← C ∪ {C_v̄}
       V ← V \ {C_v̄}
     end while
     return C
```

We give a greedy clustering algorithm (Algorithm 1) based on the Demands of the vertices to obtain a set of clusters C.

After running the algorithm, we obtain a set $\overline{V}$ with $m$ elements where $1 \leq m \leq k$. We denote $\overline{v_i}$ as the center of the corresponding cluster $C_{\overline{v_i}}$ $\forall i \in [m]$. From equations 3.2 and 3.3,

$$D(v) \leq \sum_{u \in B_r(v)} x_{uv} \leq \sum_{u \in B_r(v)} y_u \;\; \forall v \in V; \tag{3.6}$$

Coinder a vertex $\overline{v_i} \in \overline{V}$ and corresponding cluster $C_{\overline{v_i}}$. There will have to be at least $D(\overline{v_i})$ number of centers in the ball of $\overline{v_i}$ (i.e. $B_r(\overline{v_i})$). We choose $D(\overline{v_i})$ centers from $B_r(\overline{v_i})$ and open

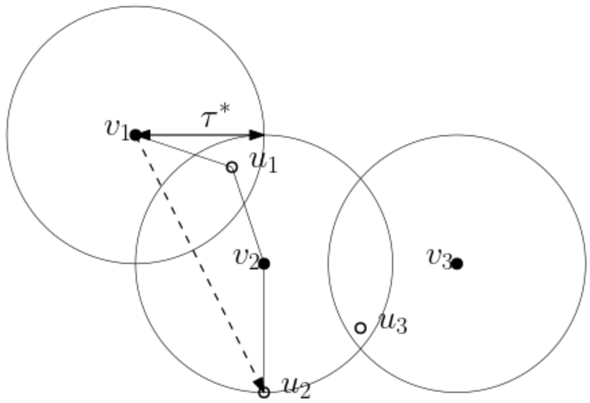


Figure 3.1: A cluster obtained from Algorithm 1

them completely. As $\overline{v_i}$ has the highest demand in the cluster $C_{\overline{v_i}}$, we can satisfy the demand $D(v) \leq D(\overline{v_i})\ \forall v \in C_{\overline{v_i}}$ by assigning $v$ to a $D(v)$ sized subset of the centers we opened in $B_r(\overline{v_i})$.

**Lemma 3.3.3** *The algorithm uses no more centers than the optimal solution.*

Each of the clusters in $C$ are disjoint. For each cluster $C_{\overline{v_i}}$ the optimal solution has to open at least $D(\overline{v_i})$ centers to satisfy the demands of the vertices in the cluster. Therefore,

$$\sum_{i=1}^{m} D(v_i) \leq OPT \tag{3.7}$$

Hence, our algorithm is at least as good as the optimal solution.

**Theorem 3.3.4** *The above algorithm returns a solution to the non-uniform demand uncapacitated K-center problem with an approximation ratio of 3.*

The cluster and reassignment of the vertices introduces an approximation factor. To satisfy the demand of the center vertex of the cluster all the centers opened a neighbours to the center vertex.

Consider a cluster $C_{v_2}$ with center vertex $v_2$. Let $v_1$ be an arbitrary vertex $v_1 \in C_{v_2}$. Let the algorithm open a center $u_2$ and assign $v_1$ to $u_2$ (see figure 3.1). As $u_2 \notin B_{\tau^*}(v_1) \cap B_{\tau^*}(v_2)$,

$$d(v_1, u_2) = d(v_1, v_2) + d(v_2, u_2) \tag{3.8}$$

$$\leq 2\tau^* + \leq \tau^* \tag{3.9}$$

$$\leq 3\tau^* \tag{3.10}$$

Hence using lemma 3.3.3, we have a **3-approximation** algorithm for the non-uniform demand uncapacitated K-center problem.

# Chapter 4

# Uncapacitated K-Center with Outliers

## 4.1 Problem

Consider the K-center problem with a generalization of having outliers. Instead of serving all the clients, given an integer $p(\leq k)$, we will find the minimum distance required by k centers to to select and serve exactly p clients. The clients disregarded are called outliers.

## 4.2 LP formation and relaxation

We consider an integer program to represent the uncapacitated K-center problem with outliers on the undirected graph $G = (V, E)$. We use the following variables in our

- Opening Variable($y_u$): an indicator variable for opening a center at $u$ where $u \in V$.
- Assignment Variable($x_{uv}$): an indicator variable denoting that $v$ is assigned to center $u$ where $u, v \in V$.
- Serving Variable($Z_v$): an indicator variable denoting that $v$ is being completely served by a center.

$$\sum_{u \in V} y_u \leq k; \tag{4.1}$$

$$x_{uv} \leq y_u \ \ \forall u, v \in V; \tag{4.2}$$

$$\sum_{u \in B_r(v)} x_{uv} \geq z_v; \tag{4.3}$$

$$\sum_{v \in V} z_v \geq p; \tag{4.4}$$

$$x, y, z \in \{0, 1\} \tag{4.5}$$

We know that 0-1 Integer programming is NP-complete, so we will consider the LP relaxation of the IP by relaxing the variables $x, y$ to

$$0 \le x, y, z \le 1 \tag{4.6}$$

## 4.3 Algorithm

In this section we describe an algorithm which gives an approximation factor of 3 for the non-uniform demand uncapacitated K-center. We guess and find out $\tau^*$ using the method described in the previous section.

**Algorithm 2 Greedy Clustering based on Center Openings**

**procedure** CLUSTER($G$)
  $C = \{\}$
  $\overline{V} = \{\}$
  $V \leftarrow$ Set of all vertices in G
  **while** $V \neq \phi$ **do**
    $\overline{v} \leftarrow \text{argmax}_{v \in V} \sum_{u \in B_r(v)} y_u$
    $\overline{V} = \overline{V} \cup \{\overline{v}\}$
    $C \leftarrow C \cup \{C_{\overline{v}}\}$
    $V \leftarrow V \setminus \{C_{\overline{v}}\}$
  **end while**
  **return** $C$

**Lemma 4.3.1** *For any cluster $C_{\overline{v}} \in C$, we can serve each vertex $v \in C_{\overline{v}}$ by the center openings within the cluster.*

Consider any arbitrary vertex $v \in C_{\overline{v}}$. From the LP, we know that the vertex has been served to an amount of $Z_v$. From equations 4.3 and 4.4,

$$z_v \le \sum_{u \in B_r(v)} x_{uv} \le \sum_{u \in B_r(v)} y_u \le \max_{v \in C_{\overline{v}}} \sum_{u \in B_r(v)} y_u \tag{4.7}$$

We reassign the the vertices of a cluster to the center openings present in the cluster itself. This gives a set of clusters where no vertex is served by a center opened in another cluster. This lets us rewrite our LP. The new LP inculcates the clusters we created from Algorithm 2.

$$\sum_{c \in C} \sum_{u \in c} \hat{y_u} \le k; \tag{4.8}$$

$$\hat{x_{uv}} \le \hat{y_u} \;\; \forall u, v \in V; \tag{4.9}$$

$$\sum_{c \in C} \sum_{u \in c} \hat{x_{uv}} \ge \hat{z_v}; \tag{4.10}$$

$$\sum_{v \in V} \hat{z_v} \ge p; \tag{4.11}$$

$$0 \le x, y, z \le 1 \tag{4.12}$$

**Definition 4.3.2** *We define $N : C \to \mathbb{Z}_{\geq 0}$ as the number of vertices in a cluster.*

Let $m = |C|$, number of the clusters in the set($k \leq m \leq n$).We order the clusters in the set C to a sequence $(c_n)$ such that $N(c_i) \geq N(c_{i+1}) \ \forall i \in [m-1]$.

Now, we choose the $k$ cluster with the highest number of vertices in the cluster. We open $k$ centers of the clusters chosen and assign every vertex in the cluster $C_{\overline{v}}$ to the cluster center $\overline{v}$.

**Lemma 4.3.3** *Consider two ordered sets $B = \{b_i : b_i \geq b_{i+1} \forall i \in [m]\}$ and $A = \{a_i : 0 \leq a_i \leq 1 \forall i \in [m]\}$ where $\sum_{i=1}^{m} a_i = l$. Then,*

$$\sum_{i=1}^{m} a_i b_i \leq \sum_{i=1}^{l} b_i \tag{4.13}$$

From Set B, we know that, $b_i \leq b_l$ for $l+1 \leq i \leq m$. Multiplying both sides by $a_i$

$$a_i b_i \leq a_i b_l \ \ \forall i \in \{l+1, l+2, ..., m\} \tag{4.14}$$

$$\tag{4.15}$$

Summing over all $i \in \{l+1, l+2, ..., m\}$,

$$\implies \sum_{i=l+1}^{m} a_i b_i \leq \sum_{i=l+1}^{m} a_i b_l \tag{4.16}$$

$$\implies \sum_{i=l+1}^{m} a_i b_i + \sum_{i=1}^{l} a_i b_i \leq \sum_{i=l+1}^{m} a_i b_l + \sum_{i=1}^{l} a_i b_i \tag{4.17}$$

We know that,

$$\sum_{i=1}^{m} a_i = l \tag{4.18}$$

$$\sum_{i=l+1}^{m} a_i = l - \sum_{i=1}^{l} a_i \tag{4.19}$$

$$\sum_{i=l+1}^{m} a_i = \sum_{i=1}^{l} (1 - a_i) \tag{4.20}$$

$$\sum_{i=l+1}^{m} a_i b_l = \sum_{i=1}^{l} (1 - a_i) b_l \tag{4.21}$$

$$(1 - a_i) b_l \leq (1 - a_i) b_i \forall i \in \{1, 2, ..., l\} \tag{4.22}$$

$$\sum_{i=1}^{l} (1 - a_i) b_l \leq \sum_{i=1}^{l} (1 - a_i) b_i \tag{4.23}$$

Using equations 4.17, 4.21 and 4.23

$$\implies \sum_{i=l+1}^{m} a_i b_i + \sum_{i=1}^{l} a_i b_i \leq \sum_{i=l+1}^{m} a_i b_l + \sum_{i=1}^{l} a_i b_i \leq \sum_{i=1}^{l} (1 - a_i) b_i + \sum_{i=1}^{l} a_i b_i \leq \sum_{i=1}^{l} b_i \tag{4.24}$$

$$\implies \sum_{i=1}^{m} a_i b_i \leq \sum_{i=1}^{l} b_i \tag{4.25}$$

**Definition 4.3.4** *We define the cluster opening function $Y : C \to \mathbb{R}_{\geq 0}$ as the sum of the total center openings in a cluster. For any cluster $C_{\overline{v}}$, $Y(C_{\overline{v}}) = \sum_{u \in C_{\overline{v}}} y_u$*

**Claim 4.3.5** *The $k$ centers chosen serve at least $p$ clients in the graph.*

From the LP equations 4.10 and 4.11,

$$p \leq \sum_{v \in V} z_v \leq \sum_{v \in V} \sum_{u \in B_r(v)} x_{uv} = \sum_{v \in V} \sum_{u:(u,v) \in E} x_{uv} \tag{4.26}$$

$$\tag{4.27}$$

As the reassignment only shifts the centers $u$ to centers inside the clusters, the total assignment in the graph remains the same. Therefore,

$$\sum_{v \in V} \sum_{u:(u,v) \in E} x_{uv} = \sum_{c \in C} \sum_{v \in c} \sum_{u \in c} \hat{x_{uv}} \tag{4.28}$$

From the LP equation 4.9

$$\hat{x_{uv}} \leq \hat{y_u} \quad \forall u, v \in V \tag{4.29}$$

$$\implies \sum_{c \in C} \sum_{v \in c} \sum_{u \in c} \hat{x_{uv}} \leq \sum_{c \in C} \sum_{v \in c} \sum_{u \in c} \hat{y_u} \tag{4.30}$$

$$\implies \sum_{c \in C} \sum_{v \in c} \sum_{u \in c} \hat{x_{uv}} \leq \sum_{c \in C} \sum_{u \in c} \hat{y_u} \sum_{v \in c} 1 \tag{4.31}$$

$$\implies \sum_{c \in C} \sum_{v \in c} \sum_{u \in c} \hat{x_{uv}} \leq \sum_{c \in C} Y(c) N(c) \tag{4.32}$$

Using equations 4.28, 4.30 and 4.32.

$$p \leq \sum_{c \in C} Y(c) N(c) \tag{4.33}$$

Once we have the ordered set in $C$, we will use the algorithm which keeps track of the cluster openings of all the clusters.

$Y_C$ is a sequence of cluster openings $Y(c)$ corresponding to cluster $c \in C$. Our algorithm 3 for mass transfer has transferred mass making 0 or more cluster openings $Y(c) = 1$ which are the beginning of the sequence followed by cluster openings $Y(c) < 1$. We open the cluster centers of all the clusters with $Y(c) = 1$. Let us say that the number of clusters with $Y(c) = 1$ in $C$ in l. We completely open $k - l(l \leq k)$ of the centers. Our required demand becomes $p' = p - \sum_{i=1}^{l} N(c_i)$. We make new sequence $(s_n)$ of the clusters just by removing the clusters whose centers are opened. For the sequence $(s_l)$, $Y(s_i) < 1 \ \forall s_i$ and the elements of sequence

**Algorithm 3** Mass Transfer

```
0: procedure MASS TRANSFER(C)
  Y_C = {}
  C' ← sequence of clusters in (c_n)
  for all c_i ∈ C' do
    if Y(c_i) > 1 And i ≤ k then
      Y(c_{i+1}) = Y(c_i) − 1
      Y(c_i) = 1
    else if Y(c_i) > 1 and i > k then
      î = min_j j such that Y(C_j) < 1
      Y(C_i) = Y(C_î)
      Y(C_î) = 1
    end if
    Y_C = Y_C ∪ {Y(C_i)}
  end for
  return Y_C
```

still follow the ordering $N(s_i) \geq N(s_{i+1})$. Using lemma 4.3.3, we get,

$$\sum_{s_i \in (s_l)} Y(s_i)N(s_i) \leq \sum_{i=1}^{l} N(s_i) \tag{4.34}$$

$$\implies \sum_{s_i \in (s_l)} Y(s_i)N(s_i) + \sum_{i=1}^{k-l} N(c_i) \leq \sum_{i=1}^{l} N(s_i) + \sum_{i=1}^{k-l} N(c_i) \tag{4.35}$$

$$\implies \sum_{c_i \in (c_n)} Y(c_i)N(c_i) \leq \sum_{s_i \in (s_l)} Y(s_i)N(s_i) + \sum_{i=1}^{k-l} N(c_i) \leq \sum_{i=1}^{l} N(s_i) + \sum_{i=1}^{k-l} N(c_i) \tag{4.36}$$

$$\implies p \leq \sum_{c_i \in (c_n)} Y(c_i)N(c_i) \leq \sum_{i=1}^{l} N(s_i) + \sum_{i=1}^{k-l} N(c_i) = \sum_{i=k-l+1}^{k} N(c_i) + \sum_{i=1}^{k-l} N(c_i) \tag{4.37}$$

$$\implies p \leq \sum_{i=1}^{k} N(c_i) \tag{4.38}$$

$$\tag{4.39}$$

The sum of vertices in the $k$ largest clusters is greater than $p$, therefore opening the cluster center in each of the $k$ largest clusters and assign every vertex in the cluster to the cluster center. Opening the $k$ centers serves $p$ vertices in the graph.

**Theorem 4.3.6** *The above algorithm returns a solution to the Uncapacitated K-center problem with Outliers with an approximation ratio of 3.*

The cluster and reassignment of the vertices introduces an approximation factor. To satisfy the demand of the center vertex of the cluster all the centers opened a neighbours to the center vertex.

Consider a cluster $C_{v_2}$ with center vertex $v_2$. Let $v_1$ be an arbitrary vertex $v_1 \in C_{v_2}$. Let the algorithm open a center $u_2$ and assign $v_1$ to $u_2$ (see figure 3.1). As $u_2 \notin B_{\tau^*}(v_1) \cap B_{\tau^*}(v_2)$,

$$d(v_1, u_2) = d(v_1, v_2) + d(v_2, u_2) \tag{4.40}$$
$$\leq 2\tau^* + \leq \tau^* \tag{4.41}$$
$$\leq 3\tau^* \tag{4.42}$$

Hence using claim 4.3.5, we have a **3-approximation** algorithm for the Uncapacitated K-center problem with Outliers.

# Chapter 5

# Capacitated K-Center with Non-Uniform Demand

## 5.1 Problem

Consider an undirected graph $G = (V, E)$ with n vertices. Given an integer $k$ with a demand function, $D : V \to \mathbb{N}$ and a capacity function $L : V \to \mathbb{Z}_{\geq 0}$ defined on $V$, capacitated k-center problem with non-uniform demand is to choose $k$ vertices to open, along with an assignment of every vertex $v$ to $D(v)$ different open centers without violating the capacity constraints of the chosen centers, such the maximum distance between a vertex and its centers is minimized.

## 5.2 Preliminary Definitions

**Definition 5.2.1** *We define the Capacity Function $L : V \to \mathbb{Z}_{\geq 0}$ which gives us a capacity constraint for every vertex in the undirected graph $G = (V, E)$ which may be chosen as a center i.e. no centeru that has been opened is assigned as a center to more than $L(u)$ vertices.*

**Definition 5.2.2** *We define the Demand function $D : V \to \mathbb{Z}_{\geq 0}$ which gives us the demand requirement of every vertex in the undirected graph $G = (V, E)$ i.e. every vertex $v$ has to be assigned $D(v) \leq K$ centers close to it.*

## 5.3 LP formation and relaxation

### 5.3.1 Program formation

We consider an integer program to represent our Non-uniform Fault-tolerant Capacitated version of the K-center problem on the undirected graph $G = (V, E)$. We use the following variables in our IP

- Opening Variable($y_u$): an indicator variable for opening a center at $u$ where $u \in V$.
- Assignment Variable($x_{uv}$): an indicator variable denoting that $v$ is assigned to center $u$ where $u, v \in V$.
- Capacity Function($L(u)$): defined above.

- Fault-Tolerance Function($D(v)$): defined above.

$$\begin{aligned}\sum_{u\in V} y_u = k;\\ x_{uv} \le y_u \;\; \forall u,v \in V;\\ \sum_{v:(u,v)\in E} x_{uv} \le L(u).y_u \;\; \forall u \in V;\\ \sum_{u:(u,v)\in E} x_{uv} \ge D(v) \;\; \forall v \in V;\\ x,y \in \{0,1\}\end{aligned}$$

We relax to IP for the valriable $x$ and $y$.

$$0 \le x, y \le 1 \tag{5.1}$$

## 5.4 Algorithm for Rounding

The solution obtained from the Linear Program has fractional values of X and Y. We will need to aggregate the openings to nearby vertices to obtain K fully open centers which would give the solution. To perform this aggregation, we first cluster the vertices such the transfer takes place only to the nearby vertices so that we don't have a high approximation factor. We then form a tree instance [1] with a sub-tree inside each cluster and aggregate inside each of the sub-trees. Any residue fractional values left from the sub-trees after aggregation are then aggregated together. In the end, we should be left a set of K fully open centers.

**Definition 5.4.1** *For a graph $G = (V, E)$ with a capacity function $L : V \to \mathbb{Z}_{\ge 0}$ and fault-tolerance function $R : V \to \mathbb{Z}_{\ge 0}$, and $y \in \mathbb{R}_+^V$, a vector $y' \in \mathbb{R}_+^V$ is a* **Distance-$r$ transfer** *if*

- $\sum_{v\in V} y_v = \sum_{v\in V} y'_v$
- $\sum_{v:d(v,U)\le r} L(v)y'_v \ge \sum_{u\in U} L(u)y_u$ *for all* $U \subseteq V$

### 5.4.1 Clustering

Khuller and Sussman [5] proposed an algorithm which partitions an undirected connected graph $G = (V, E)$, $V$ into $\{C_\nu\}_{\nu\in\Gamma}$ for some set of cluster midpoints $\Gamma \subseteq V$, such that

- we can create a tree $U = (\Gamma, F)$ such that for any edge $(u, v) \in F$, $d_G(u, v) = 3$
- for all $\nu \in \Gamma$, $N_G(\nu) \subseteq C_\nu$
- for all $u \in C_\nu$, $d_G(u, \nu) \le 2$

The algorithm, for each cluster $C_\nu$, provides an opening of at least one in the neighbourhood of $\nu$. We propose a greedy version of the algorithm for our problem.
We start with all the vertices $V$ in the graph. We select the vertex $v$with the highest demand $D(v)$. We then make a cluster $C_v$ around $v$.

**Definition 5.4.2** *We define the neighbourhood of a cluster,* $N'_G(C_v) = \{v : (u,v) \in E \ \forall u \in C_v\} \cup \{C_v\}$

**Definition 5.4.3** *We define a function* $D' : C \to \mathbb{R}_{\geq 0}$ *where* $C$ *is a set of cluster and* $D'$ *is a function which returns the maximum demand served of a vertex* $v$ *by the centers opened in the cluster. For a cluster* $C_{\overline{v}}$*,* $D'(C_{\overline{v}}) = \max_{v \in N'_G(C_{\overline{v}})} \sum_{u \in C_{\overline{v}}} x_{uv}$

---

**Algorithm 4 Greedy Clustering based on demands served by cluster**

---

0: **procedure** CLUSTER($G$)
  $\Gamma = \{\}$ {Empty Set for cluster centers}
  $C = \{\}$ {Empty Set for clusters}
  $V \leftarrow$ Set of all vertices in G
  **while** $V \neq \phi$ **do**
    $\overline{v} = \arg\max_{v \in V} D(v)$
    $\Gamma = \Gamma \cup \{\overline{v}\}$
    $C_{\overline{v}} = \{\}$
    **for all** $v \in V$ **do**
      **if** $d_G(v, \overline{v}) \leq 2$ **then**
        $C_{\overline{v}} = C_{\overline{v}} \cup \{v\}$
      **end if**
    **end for**
    $C = C \cup \{C_{\overline{v}}\}$
    $V = V \setminus \{C_{\overline{v}}\}$
  **end while**

---

The algorithm provides us with a total opening enough for maintaining fault-tolerance inside each cluster. For a cluster $C_{\overline{v}}$, where $\overline{v}$ is a cluster center, we use the LP constraints and obtain

$$\sum_{u \in C_{\overline{v}}} y_u \geq \sum_{u \in C_{\overline{v}}} x_{uv} = D'(C_{\overline{v}}) \tag{5.2}$$

$$\tag{5.3}$$

For every vertex $v \in C_{\overline{v}}$, we satisfy the demand constraint as we have at least $D'(C_{\overline{v}})$ center openings in the cluster $C_{\overline{v}}$.

### 5.4.2 Greedy Assignment

We want to aggregate the openings inside the cluster. We know that for any cluster $C_{\overline{v}}$, there will be at least $D'(C_{\overline{v}})$ openings. We open $\lfloor D'(C_{\overline{v}}) \rfloor$ highest capacitated centers completely and if there is fractional opening remaining, we open the $\lfloor D'(C_{\overline{v}}) \rfloor + 1^{th}$ to the amount of $D'(C_{\overline{v}}) - \lfloor D'(C_{\overline{v}}) \rfloor$. Once we have the centers, we do the assignment of the clients to the centers opened greedily.

We first greedily fulfill the integer demand. Once the integer demands are satisfied, we satisfy the fractional demands. Consider a cluster $C$ and store all the demands as $(D_{v_1}, D_{v_2} ... D_{v_l})$ and the store the capacities as $(C_{u_1}, C_{u_2} ... C_{u_m})$. We chose the vertex $\hat{v} = \arg\max_{v \in C} D_{\hat{v}}$. Then we choose the center $\hat{u} = \arg\max_{u \in C} C_{\hat{u}}$ and $\hat{v}$ hasn't already being assigned to it. Then we assign $\hat{v}$ to $\hat{u}$ and decrease $D_{\hat{v}}$ and $C_{\hat{u}}$ by 1. Once all the integral demands have been satisfied, we do the assignment of the demands in the same greedy manner.

**Claim 5.4.4** *The Assignment using the greedy assignment of the vertices(clients) to the centers opened in the cluster is able to satisfy the integral demands of all the vertices being served by the cluster.*

We convert each connected component into a tree with each cluster representing a node as suggested in the paper by An et al. [1]. The nodes are either leaf nodes, which have fractional cluster openings left, or are non-leaf nodes. Once we create the tree, we perform a 2-distance transfer of the cluster openings between adjacent openings. We perform transfer in a proportional manner.

**Lemma 5.4.5** *( [1]) Let $T = (V, E)$ be a tree with a capacity function $L : V \to \mathbb{Z}_{\geq 0}$ and let $Y \in [0,1]^V$ be a vector such that $y_v = 1$ for every non-leaf $v \in V$ and $\sum_v y_v \in \mathbb{Z}_{\geq 0}$. Then one can find an integral distance-2 transfer for the tree $T$.*

Using this lemma and the tree structure, we are able to get an inter-cluster transfer of distance 2. Once we get a tree and perform the distance-2 transfer, we will have served all the demands by the centers opened. The demands served by this additional 2-distance transfer may not be integral whereas we require the demands to be served integrally. But due to the transfer we know which are the potential set of centers that could serve the demands of a particular client. We leverage this fact and convert the problem to a flow problem. When we find the max-flow of the flow network we will get integral assignments. In this particular case, as we will have a flow network of the entire graph, we will get an integral assignment of all the demands of all the clients to centers. We also have a bound on the inter-cluster and intra-cluster distance transfers. Hence we will get a constant approximation factor for the solution of the tree instance.

## 5.5 The Flow Network

We make the flow network as shown in figure 5.1

**Definition 5.5.1** *We define $E : 2^V \to 2^V$ as the number of edges from the first set to the other set. $E(A_1, A_2)$ = number of edges from $A_1$ to $A_2$.*

Once we have fractional assignment, we know which vertex is assigned to which all centers. We know the assignment for the whole graph. We build a flow network of all the vertices $S$ such

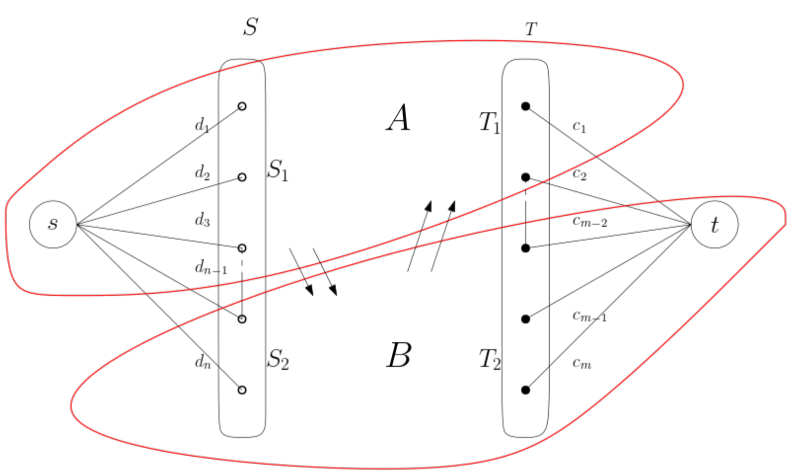


Figure 5.1: The Flow Network

that $|S| = n$ and the centers opened $T$ such that $|T| = m$. join all the vertices to the dummy vertex $s$. The capacity from vertex $s$ to $v \in S$ is the demand $d_i$ of the vertex $v$. We connect all the opened centers in $T$ centers to dummy vertex $t$ with capacity $c_i$. We make an edge between vertices $v \in S$ to centers $u \in T$ which were serving $v$ even fractionally before. We keep the capacity of each of edges as 1. To make sure that the max-flow satisfies the demands of all vertices, we need max-flow $= \sum_{i=1}^{n} d_i$. Let $Cut(A, B)$ be any arbitrary cut.

$Cut(A, B)$ = Number of Edges into $S_2$ + Number of edges from $T_1$ to $t$ + Number of edges from $S1$ to $T2$

$$Cut(A, B) = E(\{s\}, S_2) + E(T_1, \{t\}) + E(S1, T2) \tag{5.4}$$

$$\sum_{i \in S_2} d_i = E(\{s\}, S_2) \tag{5.5}$$

$$\sum_{i \in S_1} d_i = E(S_1, T_1) + E(S_1, T_2) \tag{5.6}$$

$$E(S_1, T_1) \leq \sum_{j \in T_1} c_j = E(T_1, \{t\}) \tag{5.7}$$

Using Equations 5.5, 5.6 and 5.7,

$$\sum_{i \in S} d_i = \sum_{i \in S_1} d_i + \sum_{i \in S_2} d_i \tag{5.8}$$

$$= E(S_1, T_1) + E(S_1, T_2) + E(\{s\}, S_2) \tag{5.9}$$

$$\leq E(T_1, \{t\}) + E(S_1, T_2) + E(\{s\}, S_2) \tag{5.10}$$

$$= Cut(A, B) \tag{5.11}$$

$$\tag{5.12}$$

As $\sum_{i \in S} d_i \leq Cut(A, B)$ for any arbitrary $Cut(A, B)$, $\sum_{i=1}^{n} d_i$ is the min-cut and hence the max-flow. Hence, all the demands are satisfied. We have all the assignments for the capacitated K-center with non-uniform demand.

## 5.6 Constant Approximation

**Theorem 5.6.1** *There exists a* $12-$*approximation algorithms for the capacitated K-center with non-uniform demand.*

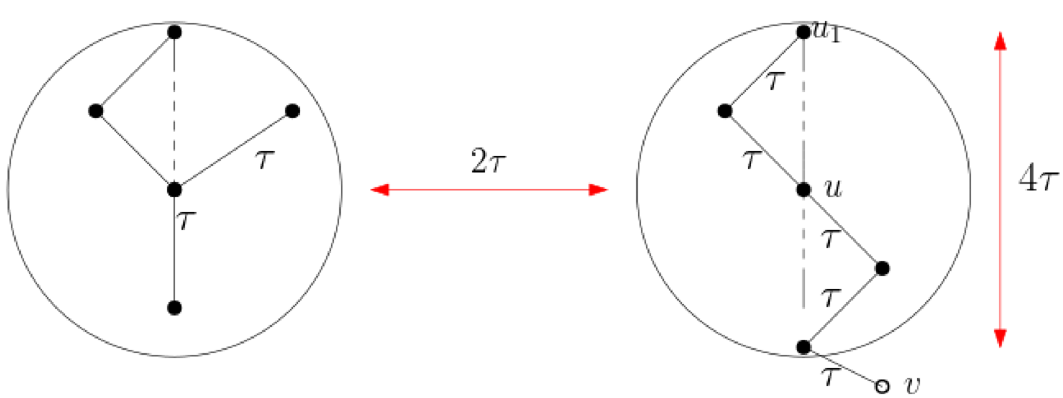


Figure 5.2: 2 Adjecent Clusters

Once we have obtained the assignments from clients to centers, we have our solution. But we need to bound the distances between the clients to the centers. We can obtain the distances by considering intra-cluster distances and inter-cluster distance. Consider the cluster with center $u$(figure 5.2). The black circles represent the opened centers. If $v$ is assigned to $u$, it brings in a distance of $d_G(v, u_1) \leq \tau + 4\tau$. This is an approximation factor of 5. So the intra-cluster approximation is 5. The inter-cluster distance between 2 adjecent cluster is a 2 approximation because of the distance-2 transfer from lemma 5.4.5. Total approximation is 2 * intra-cluster distance + inter-cluster distance. Therefore, the solution distance is $d \leq 2 \times 5\tau + 2\tau = 12\tau$. Hence we have a $12-$approximation algorithm for the capacitated K-center with non-uniform demand.

# Chapter 6

# Uncapacitated K-center with non-uniform demand and outliers

## 6.1 Problem

Consider the K-center problem with a generalization of having outliers. Instead of serving all the clients, given an integer($\leq k$), we will find the minimum distance required by k centers to select and serve exactly p clients. Each client has a non-uniform demand $d_v$ and to serve a client is to satisfy all its demands. The clients disregarded are called outliers.

## 6.2 LP formation and relaxation

We consider an integer program to represent the uncapacitated K-center problem with with non-uniform demand $d_v$ and outliers on the undirected graph $G = (V, E)$. We use the following variables in our

- Opening Variable($y_u$): an indicator variable for opening a center at $u$ where $u \in V$.
- Assignment Variable($x_{uv}$): an indicator variable denoting that $v$ is assigned to center $u$ where $u, v \in V$.
- Serving Variable($Z_v$): an indicator variable denoting that $v$ is being completely served by a center.

$$\sum_{u \in V} y_u \leq k; \tag{6.1}$$

$$x_{uv} \leq y_u \ \ \forall u, v \in V; \tag{6.2}$$

$$\sum_{u \in B_r(v)} x_{uv} \geq d_v \cdot z_v; \ \ \forall v \in V \tag{6.3}$$

$$\sum_{v \in V} z_v \geq p; \tag{6.4}$$

$$x, y, z \in \{0, 1\} \tag{6.5}$$

We know that 0-1 Integer programming is NP-complete, so we will consider the LP relaxation of the IP by relaxing the variables $x, y$ to

$$0 \leq x, y, z \leq 1 \tag{6.6}$$

## 6.3 Iterative rounding

We can observe in our LP, that $x_{uv} \leq y_u \;\; \forall u, v \in V$. As the problem is defined to be uncapacitated, we can assign as much $x_{uv}$ of cleint $v$ to the center $u$ as possible provided the inequality holds. Hence we set $x_{uv} = y_u$.

$$F(v) = \{u : x_{uv} = y_u\}$$

We now reduce the number of variables by replacing the variables $x_{uv}$ by $y_u$. We obtain the following LP

$$\sum_{u \in V} y_u \leq k; \tag{6.7}$$

$$\sum_{u \in F(v)} y_u \geq d_v \cdot z_v; \quad \forall v \in V \tag{6.8}$$

$$\sum_{v \in V} z_v \geq p; \tag{6.9}$$

$$0 \leq y, z \leq 1 \tag{6.10}$$

We will use Iterative rounding to find an integer solution for the LP. At every iteration, fix an integral value to a variable. Solve the LP using any algorithm (Simplex, Integer point). We will prove the lemma below so that we are able to fix an integral value to a variable at each iteration.

**Lemma 6.3.1** *At each iteration, when we solve the LP for uncapacitated K-center with non-uniform demand and outliers, we will have one of the trivial constraints to be tight.*

In this LP, we have a serving variable $z_v \;\; \forall v \in V$ and an assignment variable $y_u \;\; \forall u \in U$.
Total number of variables: $|V| + |U|$

We also have one constraint for inequality (6.7), one constraint for inequality (6.9) and $|V|$ many constraints for inequality type (6.8).
Total number of non-trivial constraints: $|V| + 2$

If $k > 3$, then we have more number of variables than non-trivial constraints.

For an LP to have a basic feasible solution, we need to have at least $|V| + |U|$ tight constraints i.e. at least $|V| + 3$ constraints. We have $|V| + 2$ non-trivial constraints which can be tight. This means that that there must be at least 1 trivial constraints (equation 6.10) which must be tight.

The lemma implies that one of the following must be true

- $y_u = 0$: This means we will not open the center $u$.
- $y_u = 1$: This means we will open the center $u$. For each client $v \in B_r(u)$, set $d_v \leftarrow d_v - 1$. Once a $d_{\overline{v}}$ becomes 0, we set $z_{\overline{v}} = 1$.
- $z_v = 0$: This means we will never open the client $v$.
- $z_v = 1$: This means we will open the client $v$.

In the end, we were left with fractionally open facilities which will have $\sum y_u = 2$. We were not able to solve this part and were able to find an integrality gap for the LP.

## 6.4 Integrality gap

Let us consider the following graph 6.1 to show the integrality gap in the LP for uncapacitated K-center with non-uniform demand and outliers. Let the sqaures denote the clients and the circles denote the possible centers. Nodes of $A$ form a complete bipartite graph $k_{\frac{3k}{4}-1,k}$ and the nodes of $B$ also form a complete bipartite graph $k_{k,\frac{3k}{4}-1}$. Let the clients in $A$ and $B$ have demands $d_v = \frac{3k}{4}$. The nodes in $C$ are alternating as centers(circles) and clients(squares). Let the k clients in $C$ have demands $d_v = 1$. Also the possible center $a$ in $C$ is connected to every client in $A$ and the possible center $b$ in $C$ is connected to every client in $B$. Now let us consider the problem to open $k$ centers and satisfy $\frac{5k}{3}$ clients.

**Fractional Solution**: Open any $\frac{k}{4}$ centers in each $A$ and $B$ and remaining $\frac{k}{2}$ centers in $C$ such that we open a center between clients numbered $2i-1, 2i$. Hence we will have a center open between clients 1 & 2, 3 & 4 and so on. We assign each of the $k$ clients in to each of the $\frac{k}{4}$ centers opened in $A$. We do the same in $B$. We assign the clients numbered $2i-1, 2i$ to the center opened between them. In $A$, each client is served to the demand $\frac{k}{4}$ which is equivalent to

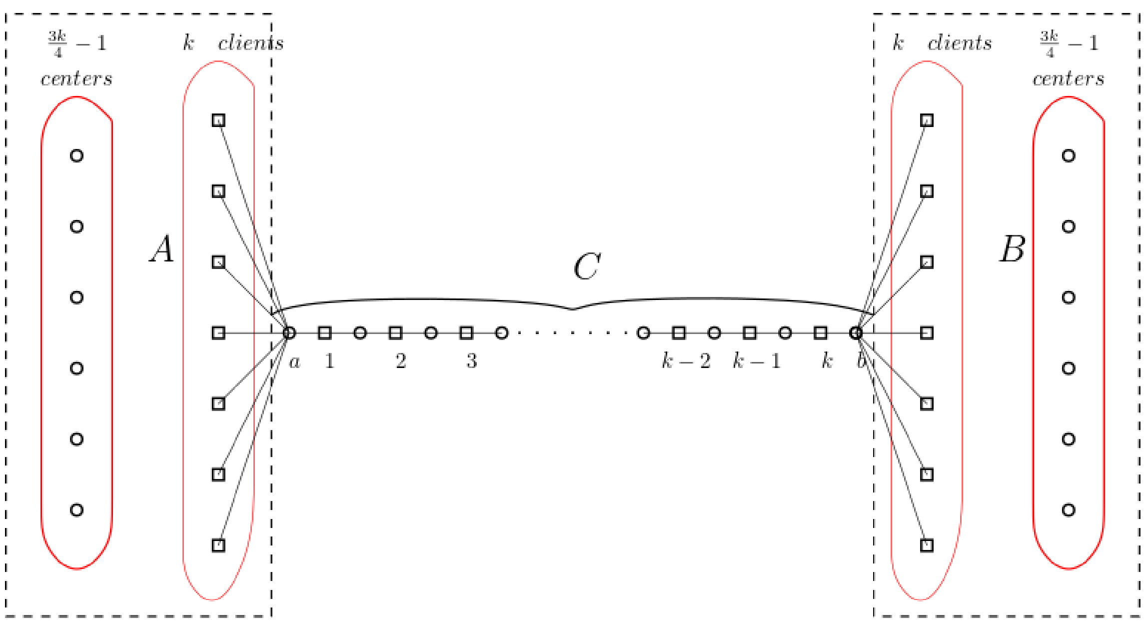


Figure 6.1: A connected graph

serving $\frac{k}{3}$ clients to their complete demand of $\frac{3k}{4}$ because $k * \frac{k}{4} = \frac{k}{3} * \frac{3k}{4}$. We have the same for $B$. As the demand of clients in $C$ are 1, they are served completely.

Total centers opened: $\frac{k}{4} + \frac{k}{4} + \frac{k}{2} = k$

Total centers opened: $\frac{k}{3} + \frac{k}{3} + k = \frac{5k}{3}$

**Integral Solution**: We will open all $\frac{3k}{4} - 1$ centers in $A$ i.e and open center $a$ in $C$. Then we will assign all the clients to each of the neighbouring $\frac{3k}{4}$ centers which will satisfy the complete demands of the the $k$ clients in $A$. We will open the remaining $\frac{k}{4}$ centers in $C$ such that we open a center between clients numbered $2i, 2i+1$. We assign the clients numbered $2i, 2i+1$ to the center opened between them. We will be able to satisfy the demands of $\frac{k}{2} + 1$ clients in $C$.

Total centers opened: $\frac{3k}{4} - 1 + 1 + \frac{k}{4} = k$

Total centers opened: $k + \frac{k}{2} + 1 = \frac{3k}{2} + 1 < \frac{5k}{3} \qquad (k \geq 7)$

To serve $\frac{5k}{3}$ integrally we will require an extra distance to serve the rest of the required clients which is dependent on $k$. Hence we observe an integrality gap which is dependent on $k$. Hence the integrality gap is unbounded.

# Chapter 7

# Future Work

- The non-uniform demand K-center with outliers has an integrality gap even for connected components. We will attempt to solve it. [6] has a method to overcome the integrality gap using a skeleton structure. We will attempt to find a structure to overcome the integrality gap for uniform demand.
- Attempt to find a constant approximation algorithm for the Capacitated K-Center with uniform demand and outliers.

# Bibliography

[1] An, H., Bhaskara, A., and Svensson, O. Centrality of trees for capacitated k-center. *CoRR abs/1304.2983* (2013).

[2] Charikar, M., Khuller, S., Mount, D. M., and Narasimhan, G. Algorithms for facility location problems with outliers. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms* (Philadelphia, PA, USA, 2001), SODA '01, Society for Industrial and Applied Mathematics, pp. 642–651.

[3] Cygan, M., Hajiaghayi, M., and Khuller, S. LP rounding for k-centers with non-uniform hard capacities. *CoRR abs/1208.3054* (2012).

[4] Khuller, S., Pless, R., and Sussmann, Y. J. Fault tolerant k-center problems. *Theor. Comput. Sci. 242*, 1-2 (July 2000), 237–245.

[5] Khuller, S., and Sussmann, Y. J. The capacitated k-center problem. *SIAM J. Discret. Math. 13*, 3 (May 2000), 403–418.

[6] Kociumaka, T., and Cygan, M. Constant factor approximation for capacitated k-center with outliers. *CoRR abs/1401.2874* (2014).