Core-Set Techniques and BQP Problem

Bolin Ding
bding3@uiuc.edu

Qiang Ma
qiangma1@uiuc.edu

Rui Yang
ruiyang1@uiuc.edu

CS598 2009 Spring

Abstract

A class of techniques to find good clustering is: given n points P in $\Re^d$, using either construction or sampling, we find small point sets S s.t. $|S| \ll n$ (say, $O(\log n)$ or some constant parameter independent on n); the task of clustering P can be (somehow) guided by the (much) smaller sets S, and results in efficient $(1+\epsilon)$-approximation algorithms. This class of techniques have been successfully applied to find $(1+\epsilon)$-approximation to the k-Center/k-Median/k-Means problems. We call this class of methods "Core-Set" techniques (a name actually given by [5]). In the first half of this report, we will survey how core-sets are used for clustering.

The Densest-kPoints problem is: given n points P, we want to select k points C from P to minimize the "variance" of these k points in C. Depending on how the "variance" is defined, we will introduce the Densest-kPoints-Center/Densest-kPoints-Median/Densest-kPoints-Mean problems, which are analogue to the k-Center/k-Median/k-Means problems. Interestingly, we will show the Binary Quadratic Programming problem (BQP) is equivalent to the Densest-kPoints-Mean problem. In the second half of this report, we also propose several approximation algorithms to solve these problems. Among them, core-set techniques lead to $(1+\epsilon)$-approximation algorithms with running time linear in the number of dimensions of points in P. Note: in the transformation from BQP to the Densest-kPoints-Mean problem, the number of dimensions could be high.

1 Introduction

Clustering problems have wide applications in the areas of data compression, machine learning, data mining, VLSI design and so on. The goal is to partition a set of objects into groups, where similar objects grouped together. Different cost functions are used to guide the clustering. Formally, given a set of n points P in $\Re^d$, we want to find a set of k points K (unnecessarily in P) s.t. $\text{cost}(P,K)$ is minimized:

In the k-Means problem, $\text{mean}_{\text{OPT}}(P,k) = \min_{K\subseteq P,|K|=k} \text{cost}(P,K)$, where

$$\text{cost}(P,K) = \text{mean}(P,K) = \sum_{p\in P} \text{dist}(p,K)^2.$$

In the k-CENTER problem, $\text{cen}_{\text{OPT}}(P,k) = \min_{K\subseteq P,|K|=k} \text{cost}(P,K)$, where

$$\text{cost}(P,K) = \text{cen}(P,K) = \max_{p\in P} \text{dist}(p,K).$$

In the k-MEDIAN problem, $\text{med}_{\text{OPT}}(P,k) = \min_{K\subseteq P,|K|=k} \text{cost}(P,K)$, where

$$\text{cost}(P,K) = \text{med}(P,K) = \sum_{p\in P} \text{dist}(p,K).$$

In the following parts, if not specified, for two points x and y in $\Re^d$, let $\text{dist}(x,y) = \|x-y\|_2$ (the euclidean distance); for two point sets X and Y, let

$$\text{dist}(X,Y) = \min_{x\in X, y\in Y} \text{dist}(x,y).$$

If $X = \{x\}$ or $Y = \{y\}$, we also write $\text{dist}(X,Y)$ as $\text{dist}(x,Y)$ or $\text{dist}(X,y)$. If $K = \{x\}$, we also simplify notations as $\text{cost}(P,x), \text{mean}(P,x), \text{cen}(P,x)$, and $\text{med}(P,x)$.

We will also study the DENSEST-kPOINTS problem for its close connection to the Binary Quadratic Programming problem, which will be demonstrated in Section 3. Formally, given a set of n points P in $\Re^d$, we want to find a set of k points $C \subseteq P$ s.t. $\text{var}(C)$ is minimized:

In the DENSEST-kPOINTS-MEAN problem, $\text{var}(C)$ is the optimal 1-means of C:

$$\text{var}(C) = \text{mean}_{\text{OPT}}(C,1) = \min_{x\in\Re^d} \text{mean}(C,x).$$

Similarly, in the DENSEST-kPOINTS-CENTER problem,

$$\text{var}(C) = \text{cen}_{\text{OPT}}(C,1) = \min_{x\in\Re^d} \text{cen}(C,x).$$

In the DENSEST-kPOINTS-MEDIAN problem,

$$\text{var}(C) = \text{med}_{\text{OPT}}(C,1) = \min_{x\in\Re^d} \text{med}(C,x).$$

Given a set of points P, let $\text{avg}(P) = \frac{\sum_{p\in P} p}{|P|}$ be the *centroid* of P. We can easily check:

Fact 1.1 $\text{mean}(P,x) = \text{mean}(P,\text{avg}(P)) + |P| \cdot \text{dist}(\text{avg}(P),x)^2$.

So we have the following fact which will be used later.

Fact 1.2 *For any set P of n points in $\Re^d$,* $\text{mean}_{\text{OPT}}(P,1) = \text{mean}(P,\text{avg}(P))$.

Outline and Contributions. In Section 2, we will do a survey on using core-sets techniques for clustering. In Section 3, we will demonstrate the connection between the Binary Quadratic Programming problem and the DENSEST-kPOINTS-MEAN problem. We propose three classes of algorithms for the three DENSEST-kPOINTS problems in Section 4. In particular, in Section 4.3, we will apply the core-sets techniques to get $(1+\epsilon)$-approximations. Finally, Section 5 concludes this report with some discussion on future works.

2 A Survey: Clustering using Core-Sets

A class of techniques to find good clustering is: given n points P in $\Re^d$, using either construction or sampling, we find small point sets S s.t. $|S| \ll n$ (say, $O(\log n)$ or some constant parameter independent on n); the task of clustering P can be (somehow) guided by the (much) smaller sets S, and results in efficient $(1+\epsilon)$-approximation algorithms.

This class of techniques have been successfully applied to find $(1+\epsilon)$-approximation to the k-Center/k-Median/k-Means problems [16, 5, 13, 17] and even when the distance measure is NOT metric [2, 1]. We call this class of methods "Core-Set" techniques (named by [5]). We will first give a survey on how core-sets is used for clustering in this section.

We first list some important papers in this topic: [16] is the first work that uses randomized core-sets to find good k-Means clustering, but it requires the clustering to be "balanced". [5] is the first one that introduces core-set techniques for k-Center and k-Median. [15] introduces core-set for projective clustering. [13] proposes faster core-set algorithms for k-Median and k-Means in data streaming environment. [12] discovers smaller core-sets for k-Median and k-Means. [4] discovers smaller core-sets for k-Center. [17] extends to techniques in [16] to find good k-Means clustering in linear time. Some other followup works include [6, 9, 8]. Two recent works [2, 1] show that core-set techniques can be applied even with some non-metric distance measures (which satisfies some properties).

In this section, we will give a survey focusing on [16] (the idea of "core-set" appeared for the first time), [17] (employed the core-sets in [16] and designed the first linear-time PTAS for k-Means), [5] (the first core-sets for k-Center and k-Median), and [2] (the first work on core-set clustering for non-metric distance measures).

2.1 Core-sets for k-Means clustering

Consider a set of n points P in $\Re^n$, one can first observe that $\text{mean}_{\text{OPT}}(P,1)$ has intrinsic statistical meanings, that is (recall Fact 1.2),

$$\frac{\text{mean}_{\text{OPT}}(P,1)}{n} = \frac{\sum_{p\in P}\|p-\text{avg}(P)\|^2}{n} \approx \frac{n-1}{n}\mathbf{Var}[X]$$

is a good estimate of $\mathbf{Var}[X]$ if P are n points generated from distribution X ($d=1$).

Ideally, sampling a set of m points $S \subseteq P$ by independent draws at random from P, $\frac{n}{m-1}\text{mean}_{\text{OPT}}(S,1)$ could be a good estimate for $\text{mean}_{\text{OPT}}(P,1)$. But this is NOT true. For example, suppose P has $n-1$ points in the same position, and one point far away from them, then $\text{mean}_{\text{OPT}}(S,1)$ is almost zero with high probability.

Fortunately, $\text{avg}(S)$ is a good estimate of $\text{avg}(P)$; we can consider using $\text{mean}(P,\text{avg}(S))$ to estimate $\text{mean}(P,\text{avg}(P)) = \text{mean}_{\text{OPT}}(P,1)$. Inaba, Katoh, and Imai [16] use this observation to design a randomized core-set algorithm for the k-Means problem.

Lemma 2.1 ([16]) *Sampling a set of m points S from a set of n points P in $\Re^d$ using m independent draws at random, for any $\delta > 0$,*

$$\|\mathrm{avg}(S) - \mathrm{avg}(P)\|^2 < \frac{1}{\delta m}\left(\frac{\sum_{p\in P}\|p - \mathrm{avg}(P)\|^2}{n}\right)$$

with probability $1 - \delta$.

Proof Sketch. Observe that

$$\mathbf{E}\left[\mathrm{avg}(S)\right] = \mathrm{avg}(P), \qquad \mathbf{E}\left[\|\mathrm{avg}(S) - \mathrm{avg}(P)\|^2\right] = \frac{1}{m}\left(\frac{\sum_{p\in P}\|p - \mathrm{avg}(P)\|^2}{n}\right)$$

and use the Markov inequality. □

Lemma 2.2 ([16]) *Sampling a set of m points S from a set of n points P in $\Re^d$ using m independent draws at random, for any $\delta > 0$,*

$$\mathrm{mean}(P, \mathrm{avg}(S)) < \left(1 + \frac{1}{\delta m}\right)\mathrm{mean}(P, \mathrm{avg}(P)) = \left(1 + \frac{1}{\delta m}\right)\mathrm{mean}_{\mathrm{OPT}}(P, 1)$$

with probability $1 - \delta$.

Proof Sketch. Recall Fact 1.2,

$$\mathrm{mean}(P, \mathrm{avg}(S)) = \mathrm{mean}(P, \mathrm{avg}(P)) + |P| \cdot \|\mathrm{avg}(P) - \mathrm{avg}(S)\|^2,$$

and use Lemma 2.2. □

2.1.1 Balanced 2-clustering

We first focus on the case that $k = 2$ and assume that clustering are balanced enough. More formally, suppose $K^* = \{c_1^*, c_2^*\}$ is the optimal solution to the 2-Means problem, and let $P(c_i^*)$ be the points which are more closer to c_i^* than c_j^* $(j \neq i)$ in a point set P, we say that P is $f(m)$-*balanced* if there exists an optimal solution K^* with

$$\frac{\min\{|P(c_1^*)|, |P(c_2^*)|\}}{n} \geq \frac{f(m)}{m},$$

and K^* is called an $f(m)$-*balanced optimal 2-Means clustering*.

Given an $f(m)$-balanced point set P, we can expect that, in a random sampling S from P, the numbers of points from $P(c_1^*)$ and $P(c_2^*)$, respectively, are also balanced. Define $S(c_i^*)$ similarly. More formally, we can prove the following lemma using the Chernoff bound.

Lemma 2.3 ([16]) *Suppose $K^* = \{c_1^*, c_2^*)$ is an $f(m)$-balanced optimal 2-Means clustering of P, where $f(m) \geq \log m$. Then for any $\beta > 0$, with probability $1 - \frac{2}{m^{\beta^2/2}}$,*

$$\frac{\min\{|S(c_1^*)|, |S(c_2^*)|\}}{m} \geq (1-\beta)\frac{\min\{|P(c_1^*)|, |P(c_2^*)|\}}{n} \geq (1-\beta)\frac{f(m)}{m} \geq (1-\beta)\frac{\log m}{m}.$$

If $S(c_i^*)$ is large enough, as Lemma 2.2, since $S(c_i^*)$ is a random sampling from $P(c_i^*)$, we can use $\text{mean}(P(c_i^*), \text{avg}(S(c_i^*)))$ to approximate $\text{mean}(P(c_i^*), \text{avg}(P(c_i^*)))$. The only problem is: from the point set S, we do not know which part is $S(c_1^*)$ and which part is $S(c_2^*)$. However, since $|S|$ is independent on $|P| = n$ or d, we can simply guess all the (linearly separable) partitions of S. Inaba *et al.* give the following algorithm.

BALANCED2CLUSTERING($P \subseteq \Re^d$):
Sample a set of m points S from P using m independent draws at random.
For every partition of S: $S = S_1 \cup S_2$, compute $c_1 = \text{avg}(S_1)$ and $c_2 = \text{avg}(S_2)$.
Output $K = \{c_1, c_2\}$ with the minimum $\text{mean}(P, K)$ among all (c_1, c_2)'s generated above.

Note $\text{mean}(P, \{c_1, c_2\}) = \text{mean}(P(c_1), c_1) + \text{mean}(P(c_2), c_2)$. Using Lemma 2.2 and 2.3:

Theorem 2.4 ([16]) *Suppose the point set P is $f(m)$-balanced with $f(m) \geq \log(m)$. Then, the* BALANCED2CLUSTERING *algorithm finds an $(1 + \frac{1}{\delta(1-\beta)f(m)})$-approximation for the 2-Means problem with probability $1 - \delta - \frac{2}{m^{\beta^2/2}}$ in $O(\min\{nm^d, nd2^m\})$ time.*

2.1.2 The general case

Although the BALANCED2CLUSTERING algorithm can be generalized to handle the case $k > 2$, the "$f(m)$-balanced" assumption makes it insufficient to find good clustering in general. Now we present the algorithm by Kumar, Sabharwal, and Sen [17] for finding good k-Means clustering use the same core-set, the sampling set S, but of course, a different (and more complicated) clustering algorithm.

Again, we focus on 2-Means clustering first. Recall $P(c_i^*)$ are the points which are more closer to c_i^* than c_j^* $(j \neq i)$ in a point set P, for $i = 1, \ldots, k$. Algorithms in [17] are based on the following observation: (i) if P is very "unbalanced" (i.e. one of $|P(c_1^*)|$ and $|P(c_2^*)|$ is very small), then we can reduce 2-Means clustering to 1-Means clustering; (ii) at least one of $|P(c_1^*)|$ and $|P(c_2^*)|$ is larger than $|P|/2$. Observation (i) leads to the following definition. (Recall $P(c_i^*)$ is the set of points which are more closer to c_i^* than c_j^* $(j \neq i)$.)

Definition 2.5 (Mean-Reducible) *We say the point that set P is (k, ϵ)-irreducible if $\text{mean}_{\text{OPT}}(P, k-1) \geq (1 + 32\epsilon)\text{mean}_{\text{OPT}}(P, k)$. Otherwise, we say it is (k, ϵ)-reducible.*

Let $\alpha = \epsilon/64$. We assume P is $(2, \alpha)$-irreducible. Otherwise, directly from the definition, we can get an $(1 + \epsilon)$-approximation to 2-Means using 1-Means.

We sample, again, a point set S of size $O(\frac{1}{\epsilon})$, say $|S| = \frac{4}{\beta\epsilon}$, from P. Consider an optimal 2-Means clustering $K = \{c_1^*, c_2^*\}$ for P. W.l.o.g., assume that $|P(c_1^*)| \geq |P|/2$. Using the

similar (weaker) idea of Lemma 2.3, we can show that $|S(c_1^*)| \geq \frac{2}{\epsilon}$ with high probability. We can guess $S(c_1^*)$ (to be precise, its subset of size $\frac{2}{\epsilon}$—cycling through all such subsets of S), and from Lemma 2.2, we can assume that we have found $c_1 = \text{avg}(S(c_1^*))$ s.t.

$$\text{mean}(P(c_1^*), c_1) \leq (1+\alpha)\text{mean}(P(c_1^*), c_1^*).$$

Lemma 2.6 ([17]) *Let* $\text{dist}(c_1^*, c_2^*) = t$. *We have* c_1^* *and* c_1 *are closed:* $\text{dist}(c_1^*, c_1) \leq t/4$.

Proof. Otherwise ($\text{dist}(c_1^*, c_1) > t/4$), from Fact 1.1, we have

$$\alpha \cdot \text{mean}(P(c_1^*), c_1^*) \geq \text{mean}(P(c_1^*), c_1) - \text{mean}(P(c_1^*), c_1^*) = |P(c_1^*)|\text{dist}(c_1^*, c_1)^2 \geq \frac{t^2|P(c_1^*)|}{16};$$

and thus, $\text{mean}(P(c_1^*), c_2^*) = \text{mean}(P(c_1^*), c_1^*) + |P(c_1^*)|t^2 \leq (1+16\alpha)\text{mean}(P(c_1^*), c_1^*)$. So, $\text{mean}(P, c_2^*) \leq (1+16\alpha)\text{mean}(P, \{c_1^*, c_2^*\})$, i.e., P is $(2, \alpha)$-reducible (contradiction). □

The previous lemma implies the ball $\mathcal{B}(c_1, t/4)$ (of radius $t/4$ centered at c_1) is contained in $\mathcal{B}(c_1^*, t/2)$. So, $\mathcal{B}(c_1, t/4) \cap P \subseteq P(c_1^*)$. We are now interested in c_2^* and $P(c_2^*)$. So we expect $P' = P - \mathcal{B}(c_1, t/4)$ has a good fraction of points from $P(c_2^*)$. Actually, it is true.

Lemma 2.7 ([17]) *Let* $P_1' = P(c_1^*) - \mathcal{B}(c_1, t/4)$. *Then* $P' = P_1' \cap P(c_2^*)$ *and* $|P(c_2^*)| \geq \alpha|P_1'|$.

Proof. The intuition is: if $P(c_2^*)$ is small in P' still, c_1^* could be a good 1-Means solution.

Suppose $|P(c_2^*)| \leq \alpha|P_1'|$. Notice $\text{mean}(P(c_1^*), c_1) \geq \text{mean}(P_1', c_1) \geq \left(\frac{t}{4}\right)^2 |P_1'| = t^2|P_1'|/16$. Since $\text{mean}(P(c_1^*), c_1) \leq (1+\alpha)\text{mean}(P(c_1^*), c_1^*)$, we have

$$t^2|P(c_2^*)| \leq \alpha t^2|P_1'| \leq 16\alpha(1+\alpha)\text{mean}(P(c_1^*), c_1^*)$$

and thus

$$\text{mean}(P, c_1^*) = \text{mean}(P(c_1^*), c_1^*) + \text{mean}(P(c_2^*), c_2^*) + t^2|P(c_2^*)| \leq (1+32\alpha)\text{mean}(P, \{c_1^*, c_2^*\}),$$

i.e., P is $(2, \alpha)$-reducible (contradiction). □

So the algorithm proceeds as follows: we sample a point set S' of size $O(\frac{1}{\alpha^2})$ from P'. Again, we can show $|S'(c_2^*)| \geq \frac{2}{\alpha}$ with high probability. So we cycle through all subsets of size $\frac{2}{\alpha}$, and can find a point c_2 s.t. $\text{mean}(P(c_2^*), c_2) \leq (1+\alpha)\text{mean}(P(c_2^*), c_2^*)$. Therefore, $K = \{c_1, c_2\}$ is an $O(1+\alpha)$-approximation to the 2-Means problem.

The only problem is that we do not know $\text{dist}(c_1^*, c_2^*) = t$, and thus cannot identify P' (as well as sample S' from P'). However, we can find c_1 without the knowledge about t, and guess the parameter i s.t. $\frac{n}{2^i} \leq |P'| \leq \frac{n}{2^{i-1}}$. Let P'' be the $\frac{n}{2^{i-1}}$ farthest points from c_1 in P, and we know $P'' \supseteq P', |P''| \leq 2|P'|$. We increase the sample size $|S'|$ (from P'') a bit, and can get the same result. This algorithm needs $O(\log n)$ iterations (guesses i). More involved analysis shows the running time is linear in n ($|P''|$ decreases by half in every iteration).

Theorem 2.8 ([17]) *Given a point set* P *of size* n *in* $\Re^d$, *we can find* $(1+\epsilon)$*-approximation to the optimal 2-Means with constant probability in time* $O(2^{(1/\epsilon)^{O(1)}}nd)$. *Similar idea applies to k-Means clustering with running time* $O(2^{(k/\epsilon)^{O(1)}}nd)$.

2.2 Core-sets for k-Center clustering

Similar to the core-set for k-Means, given a set of n points P in $\Re^d$, we want to find a small set $S \subseteq P$ s.t. the 1-Center of S can be used to approximate the 1-Center of P (somehow). Because of some geometric properties, the core-set S for k-Center is stronger than the one for k-Means (recall $\text{mean}_{\text{OPT}}(S,1)$ cannot be used to estimate $\text{mean}_{\text{OPT}}(P,1)$, but, here, $\text{cen}_{\text{OPT}}(S,1)$ approximates $\text{cen}_{\text{OPT}}(P,1)$); however, S here cannot be obtained by sampling from P (while most samplings are good), but it can be obtained in a constructive way.

For a point set P, let $c_P = \arg\min_x \text{cen}(P,x)$, i.e., $\text{cen}_{\text{OPT}}(P,1) = \text{cen}(P,c_P)$ (c_P is the minimum enclosing ball of P). c_P could be found using convex programming techniques. We also write $\text{cen}_{\text{OPT}}(P,1)$ as r_P (the radius of the minimum enclosing ball of P).

Lemma 2.9 ([5]) *There is a subset of points $S \subseteq P$ with $|S| = O(\frac{1}{\epsilon^2})$ s.t. $\text{cen}_{\text{OPT}}(S,1) \geq \text{cen}_{\text{OPT}}(P,1)/(1+\epsilon)$ and*

$$\text{cen}(P,c_S) \leq (1+\epsilon)\text{cen}(S,c_S) \leq (1+\epsilon)\text{cen}(P,c_P) = (1+\epsilon)\text{cen}_{\text{OPT}}(P,1).$$

Proof. We start constructing S with $S_0 = \{x,y\}$ s.t. $x,y \in P$ and $\text{dist}(x,y) \geq r_P$. x is the furthest point away from y in P. Clearly, $r_P/2 \leq r_{S_0} \leq r_P$.

There are two cases:

(i) If there is no point $p \in P$ s.t. $\text{dist}(p,c_{S_i}) \geq (1+\epsilon)r_{S_i}$, it is done: $S = S_i$.

(ii) If there is a point $p \in P$ s.t. $\text{dist}(p,c_{S_i}) \geq (1+\epsilon)r_{S_i}$, set $S_{i+1} = S_i \cup \{p\}$ and $i = i+1$.

We claim $r_{S_{i+1}} \geq (1+\epsilon^2/16)r_{S_i}$ and thus the above iteration can repeat at most $O(\frac{1}{\epsilon^2})$ times. Finally, S has size $O(\frac{1}{\epsilon^2})$ and satisfies the desired property.

This claim can be proved as follows.

If $\text{dist}(c_{S_i}, c_{S_{i+1}}) < \epsilon r_{S_i}/2$, then by triangle inequality,

$$r_{S_{i+1}} \geq \text{dist}(p,c_{S_{i+1}}) \geq \text{dist}(p,c_{S_i}) - \text{dist}(c_{S_i},c_{S_{i+1}}) \geq \left(1+\frac{\epsilon}{2}\right) r_{S_i}.$$

If $\text{dist}(c_{S_i}, c_{S_{i+1}}) \geq \epsilon r_{S_i}/2$, then let H be the $(d-1)$-dim hyperplane that passes through c_i and is orthogonal to $c_{S_i}c_{S_{i+1}}$. Let H^- be the open half-space having p inside. Then we know there is a point $x \in S_i$ in the closed half-space $\Re^d - H^-$ s.t. $\text{dist}(c_{S_i},x) = r_i$ (note the minimum enclosing ball of S_i centered at c_{S_i}) [10]. Therefore,

$$r_{S_{i+1}} \geq \text{dist}(c_{S_{i+1}},x) \geq \sqrt{r_{S_i}^2 + \frac{\epsilon^2}{4} r_{S_i}^2} \geq \left(1+\frac{\epsilon^2}{16}\right) r_{S_i}.$$

So the proof is completed. □

The algorithm of 2-Center clustering is as follows. Suppose P is partitioned into two sets X and Y in the optimal solution. We start from two empty point sets S_X and S_Y as the core-sets for X and Y, respectively. In each of the following iterations, we pick a point p furthest away from $S_X \cup S_Y$. From a guessing oracle, if $p \in X$, then put p into S_X; otherwise, put p into S_Y. From the above lemma, after $O(1/\epsilon^2)$ iterations, we can get an $O(1+\epsilon)$-approximation for 2-Center clustering ($K = \{c_{S_X}, c_{S_Y}\}$).

To remove the guessing oracle, we enumerate all the possibilities, which needs $2^{O(1/\epsilon^2)}$. This algorithm can be also extended for general k, and the running time turns to be $k^{O(k/\epsilon^2)}$.

Theorem 2.10 ([5]) *For any point set P with size n in $\Re^d$ and $0 < \epsilon < 1$, an $(1+\epsilon)$-approximation of k-Center clustering for P can be found in $2^{O(k \log k/\epsilon^2)}nd$ time.*

Smaller core-set of size $O(1/\epsilon)$ for the k-Center clustering is found in [4].

2.3 Core-sets for k-Median clustering

The construction of core-sets of k-Median clustering is more complicated than the ones of k-Means and k-Center clustering. Badoiu, Har-Peled, and Indyk proposed an randomized construction in [5]. Note: the size of core-set for k-Median might be sublinearly dependent on n (say $O(\log n)$), but is still small enough for designing efficient algorithms on it.

For a set of n points P in $\Re^d$, let $\mathrm{AvgMed}(P,k) = \mathrm{med}_{\mathrm{OPT}}(P,k)/|P|$, which can be interpreted as the average "radius" of the k-Median clustering.

Specifically, let $K^* = \{c^*\}$ be the optimal solution to 1-Median clustering $\mathrm{med}_{\mathrm{OPT}}(P,1)$ on P. The following lemma says, we can find an $(1+\epsilon)$-approximation c' for $\mathrm{med}_{\mathrm{OPT}}(P,1)$ in a space spanned by a small number of samples from P (also, c' and c^* are closed).

Lemma 2.11 ([5]) *Let H be a random sample of $O(1/\epsilon^3 \log 1/\epsilon)$ points from P. With constant probability, these two events happen: (i) The flat spanned by H, $\mathrm{span}(H)$, contains a $(1+\epsilon)$-approximation 1-Median, c', for P, and (ii) H contains a point in distance $\leq 2\mathrm{AvgMed}(P,1)$ from the center of the optimal solution, c^*.*

Proof Sketch. We skip the detailed proof here. But the main idea is similar to the one in Lemma 2.9. First, from the definition, we immediately have $\mathbf{E}[\mathrm{dist}(s_i, c^*)] = \mathrm{AvgMed}(P,1)$, if s_i is a point sampled uniformly from P (so (ii) is true). Let F_i be the flat spanned by the first i samples $s_1, s_2, \ldots, s_i$, i.e., $\mathrm{span}(\{s_1, \ldots, s_i\})$, and c' be the projection of c^* on F_i. It can be shown $\mathrm{dist}(c', c^*)$ shrinks very quickly as long as there are enough points in P NOT closed to F_i. Finally, either $\mathrm{dist}(c', c^*)$ becomes small enough or most points are closed to F_i; in both cases, c' could be a good approximation to c^*, and $H = \{s_1, s_2, \ldots\}$. □

Before presenting the construction of core-sets for k-Median clustering, we assume:

(a) The distance between any two points in P is at least one;
(b) The optimal k-Median cost $\mathrm{med}_{\mathrm{OPT}}(P,k)$ is at most n^b for some $b = O(1)$.

If (a) or (b) is not true, we cover space by a grid of size $L\epsilon/(5nd)$, and snap points of P to this grid, where L satisfying $L/2 \leq \mathrm{med}_{\mathrm{OPT}}(P,k) \leq nL$ can be find using an 2-approximation algorithm for the k-Center clustering. The cost of any k-Median clustering in the new point set differs at most a factor of $(1+\epsilon/5)$ from the same one of P.

Now we present the construction of core-sets for k-Median clustering. The idea is based on (i) and (ii) in Lemma 2.11. Let's first assume we have found t s.t. $t/2 \leq \mathrm{AvgMed}(P,1) \leq t$. Clearly, t can be found by checking $t = 2^i$ for $i = 0, 1, \ldots, O(\log n)$ because of (b).

Let H be a random sample of $O(1/\epsilon^3 \log 1/\epsilon)$ points from P. As in Lemma 2.11 (i), c' (the projection of c^* on span(H)) is a good approximation to the 1-Median solution for P. So our goal is to find a small set of points $S(P,H)$ (core-set), s.t. some of them is closed to c' and thus could be used to approximate 1-Median solution for P. From Lemma 2.11 (ii), some point in H is in distance $\leq 2t$ from c^* (and thus $\leq 2t$ from c'). Therefore, we construct a grid near each point of H to locate c', and the vertices of the grids form the set $S(P,H)$.

A bit more formally, let $\mathcal{G}_p(t)$ be a grid of side length $O(\epsilon t/|R|)$ centered at p on H, and let $\mathcal{B}(p,2t)$ be a ball of radius $2t$ centered at p. Let $S'(p,t) = \mathcal{G}_p(t) \cap \mathcal{B}(p,2t)$. Clearly, if $\text{dist}(p,c^*) \leq 2t$, then c' falls into $\mathcal{B}(p,2t)$, and thus some point in $S'(p,t)$ can be used as an $(1+\epsilon)$-approximation to the 1-Median solution for P.

Finally, $S(P,H) = \bigcup_{i=0}^{O(\log n)} \bigcup_{p\in H} S'(p,2^i)$, and $|S(P,H)| = O\left(2^{O(1/\epsilon^4)} \log n\right)$.

Lemma 2.12 ([5]) *Let H be a random sample of $O(1/\epsilon^3 \log 1/\epsilon)$ points from P. One can compute a point set $S(P,H)$ of size $O\left(2^{O(1/\epsilon^4)} \log n\right)$, s.t. with high probability (over the choice of H), there is a point $q \in S(P,H)$ s.t. $\text{med}(P,q) \leq (1+\epsilon)\text{med}_{\text{OPT}}(P,1)$.*

Using the ideas similar to the algorithm for k-Means described in Section 2.1.2, one may obtain "efficient" $(1+\epsilon)$-approximation algorithms for k-Median clustering.

Theorem 2.13 ([5]) *Given a point set P of size n in $\Re^n$, we can find $(1+\epsilon)$ approximation to the optimal 2-Median with high probability in time $O(2^{(1/\epsilon)^{O(1)}} d^{O(1)} n \log^{O(1)} n)$. Similar idea applies to k-Median clustering with running time $O(2^{(k/\epsilon)^{O(1)}} d^{O(1)} n \log^{O(k)} n)$.*

2.4 Core-sets for non-metric distance clustering

Recently, Ackermann, Blömer, and Sohler [2] and their followup work [1] extend the core-set techniques to the non-metric distance clustering. They show that if a (maybe *non-metric*) distance measure is $[\gamma,\delta]$-*sampleable*, the algorithm in [17] (see Section 2.1.2 here) can be adapted to find $(1+\epsilon)$-approximation to the *generalized* k-Median *problem* in linear time.

Note the only different between the generalized k-Median problem and the k-Median problem is that, in the generalized k-Median problem, *we do NOT require the distance measure to be a metric.* We allow $\text{dist}(x,y) \neq \text{dist}(y,x)$ and $\text{dist}(x,y)+\text{dist}(y,z) < \text{dist}(x.z)$, but only require $\text{dist}(x,y) = 0 \Leftrightarrow x = y$. So the k-Means problem is a special case of the generalized k-Median problem if the distance measure is defined to be $\text{dist}(x,y) = \|x-y\|_2^2$.

The algorithm in [2] is nearly identical to the one in [17]. But, a significant difference between Ackermann *et al.*'s work and [17] is that the analysis of Ackermann *et al.*'s algorithms does NOT depend on the symmetry or the triangle inequality of distance measure, while previous works, like [17], do. Also, they discuss which (non-metric) distance measures are $[\gamma,\delta]$-sampleable, and construct core-sets for these measures.

Below, let c_P be the optimal 1-Median of any point set P, i.e. $\text{med}(P,c_P) = \text{med}_{\text{OPT}}(P,1)$.

We first state the main result of [2]. Note $\text{dist}(x,y)$ is unnecessarily $\|x-y\|_2$ now.

Theorem 2.14 *Given an integer k and any $\epsilon < 1$. Assume that for $\delta < 1$ and $\beta = \epsilon/3$, distance measure* $\mathrm{dist}(\cdot,\cdot)$ *satisfies:*

(a) For every finite point set S, an optimal 1-Median c_S, i.e. $\mathrm{med}(S, c_S) = \mathrm{med}_{\mathrm{OPT}}(S, 1)$, *can be computed in time depending only on $|S|$.*

(b) There exists a constant $m_{\gamma,\delta}$ such that for every point set P of size n and for every uniform sample multiset $S \subseteq P$ of size $m_{\gamma,\delta}$, an optimal 1-Median c_S of S satisfies

$$\mathbf{Pr}\left[\mathrm{med}(P, c_S) \leq (1+\gamma)\mathrm{med}_{\mathrm{OPT}}(P, 1)\right] \geq 1 - \delta.$$

Then there exists an algorithm that with constant probability returns an $(1+\epsilon)$-approximation of the k-Median *problem w.r.t.* $\mathrm{dist}(\cdot,\cdot)$ *for input point set P of size n in time $O(n2^{(\frac{k}{\epsilon})^{O(1)}})$.*

(b) in the above theorem is called *superset sampling* or *core-set sampling* (for S is the so-called "core-set"). Their analysis is even simpler than the one in [17] (Section 2.1.2 here).

Let's restate the algorithm for $k = 2$. Let $K^* = \{c_1^*, c_2^*\}$ be the optimal 2-Median, and $P_i^* = P(c_i^*)$ be the cluster containing c_i^* with $|P_1^*| \geq \alpha|P|$ ($\alpha > 1/4$).

1. (Superset sampling) Obtain c_1 from P with $\mathrm{med}(P_1^*, c_1) \leq (1+\gamma)\mathrm{med}_{\mathrm{OPT}}(P_1^*, 1)$.
2. Let N be the smallest subset of closest points from P towards c_1 s.t. for the remaining points $R = P/N$, we have $|P_2^* \cap R| \geq \alpha|R|$. Assign N to c_1. Note: $P_2^* \cap R = P_2^*/N$.
3. (Superset sampling) Obtain c_2 from R with $\mathrm{med}(P_2^* \cap R, c_2) \leq (1+\gamma)\mathrm{med}_{\mathrm{OPT}}(P_2^* \cap R, 1)$.
4. Use $K = \{c_1, c_2\}$ as a 2-Median solution.

In 1 above, to obtain c_1, we take a sample multiset S of size $m_{\gamma,\delta}$ from P and enumerate all the $O(\frac{m_{\gamma,\delta}}{\alpha})$-subsets of S. Similar guessing oracle is used in 3. So if N is known, the running time is $O\left(n2^{(\frac{2}{\alpha}m_{\gamma,\delta})^{O(1)}}\right)$.

Of course, N is unknown. So N is approximated by partitioning P into $N^{(1)}$, $N^{(2)}$, ..., $N^{(\lceil \log n \rceil)}$. Here, $N^{(1)}$ is the $n/2$ closest points to c_1; $N^{(2)}$ is the next $n/4$ closest points to c_1; $N^{(3)}$ is the next $n/8$ closest points to c_1; .... Let $R^{(j)} = P/\bigcup_{i=1}^{j} N^{(i)}$, and let v be the minimal value s.t. $|P_2^* \cap R^{(v)}| \geq \alpha|R^{(v)}|$. We will approximate $N = N^{(1)} \cup N^{(2)} \cup \ldots \cup N^{(v)}$.

Then,

$$\mathrm{med}(P, K) \leq \mathrm{med}(P_1^*, c_1) + \mathrm{med}(P_2^* \cap N, c_1) + \mathrm{med}(P_2^*/N, c_2).$$

Using the two claims: (i) $\mathrm{med}(P_2^* \cap N, c_1) \leq 8\alpha \cdot \mathrm{med}(P_1^*, c_1)$; (ii) $\mathrm{med}(P_1^*, c_1) \leq (1+\gamma) \cdot \mathrm{med}(P_1^*, c_1^*)$, and (iii) $\mathrm{med}(P_2^*/N, c_2) \leq (1+\gamma) \cdot \mathrm{med}(P_2^*/N, c_2^*)$.

(ii) and (iii) are because of the superset sampling technique. (i) is non-trivial but it is mainly because there are fewer points in $N^{(j)}$ from P_2^* ($\leq 2\alpha$) than from P_1^* ($> 1 - 2\alpha$), if $j \leq v$. Then we can conclude:

$$\mathrm{med}(P, K) \leq (1+8\alpha)(1+\gamma)\mathrm{med}(P_1^*, c_1^*) + (1+\gamma)\mathrm{med}(P_2^*/N, c_2^*) \leq (1+8\alpha)(1+\gamma)\mathrm{med}_{\mathrm{OPT}}(P, 2).$$

In the case that N does not exist, or more precisely, $v = \lceil \log n \rceil$, we end up with a single point $R^{(v)} = \{q\}$. Let q itself forms a cluster with cost 0, and the above analysis is still valid. Therefore, this algorithm gives an $(1 + 8\alpha)(1 + \gamma))$-approximation.

Constructing core-sets for non-metric distance. Given the algorithm above, the rest question is for which non-metric distance measures, the *superset sampling* technique is valid for finding good core-sets. [2, 1] introduce such sampling techniques for non-metric distance measures like the Kullback-Leibler divergence, Mahalanobis distance, Bregman divergance, etc. So the above algorithm is valid for a broad class of distance definitions.

3 Densest k-Points v.s. Binary Quadratic Programing

The formulation of Binary Quadratic Programming problem (BQP) is the following:

$$\begin{aligned} \max \quad & x^T Q x \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = k \end{aligned}$$

$$Q = [q_{ij}]_{n\times n}, \quad x = [x_i]_{1\times n}, \quad x_i \in \{0, 1\}.$$

The BQP and many of its special cases as well are NP-hard problems. Numerous hard combinatorial optimization problems can be formulated as BQP problem, including capital budgeting and financial analysis, traffic message management and also some graph problems like max cut problem, max clique problem and maximum independent set problem [19].

The main reason leading us to study the Densest-kPoints-Mean problem is its interesting connection to the Binary Quadratic Programming problem. In the following subsection we will show the equivalence between these two problems.

3.1 Equivalence between BQP and 1-mean problem

Without loss of generality, we narrow down the choice of Q into a small set. We claim that we only need to consider the case that Q is positive definite and the diagonal elements are identical. First we claim that we can take matrix Q that has all zero diagonal elements. This is because (let Q_{off} be the matrix replacing all diagonal elements of Q with 0, and diag(Q) be the vector formed by diagonal elements of Q):

$$\begin{aligned} x^T Q x \quad &= x^T Q_{\text{off}} x + \text{diag}(Q)^T x \quad \text{(note } x \cdot x = x \text{ since it is binary)} \\ &= \begin{pmatrix} x^T & 1 \end{pmatrix} \begin{pmatrix} Q_{\text{off}} & \text{diag}(Q) \\ \text{diag}(Q)^T & 0 \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix} = \begin{pmatrix} x^T & 1 \end{pmatrix} Q_+ \begin{pmatrix} x \\ 1 \end{pmatrix}. \end{aligned}$$

The enlarged matrix Q_+ has desired property. Secondly, to make Q positive definite, we add something on the diagonal part. In our problem all elements in Q is nonnegative. Let $\lambda = \max_j(\sum_{i=1}^{n} q_{ij})$, we define

$$\bar{Q} = Q + \lambda I$$

It is clear that the matrix $\bar{Q}$ is positive definite since it's diagonal dominant. Hence we have the following claim

Claim 3.1 *Considering Q as a positive definite and identical diagonal elements matrix is enough.*

Since Q is positive definite, we can operate Cholesky decomposition on Q, i.e., $Q = V^T V$ where V is a lower triangular matrix. The objective function thus turns to $x^T Q x = x^T V^T V x = \| \sum_{i=1}^{n} x_i V_i \|^2$. We also have $\|V_i\|^2 = c$ where c is the diagonal element of Q.

If we consider the column vector V_i as a point in $\Re^n$, then all points $P = \{V_1, V_2, \ldots, V_n\}$ are on a surface of a ball in $\Re^n$. The objective function is hence transformed into:

$$\max \| \sum_{i=1}^{n} x_i V_i \|^2 \iff \max_{C \subseteq P:\ |C|=k} \| \sum_{V_i \in C} V_i \|^2,$$

i.e., finding a set of points $C \subseteq P$ s.t. $|C| = k$ (for $\sum_i x_i = k$) to minimize $\| \sum_{V_i \in C} V_i \|^2$. It can be further transformed into:

$$\begin{aligned}
\max_{C \subseteq P:\ |C|=k} \| \sum_{V_i \in C} V_i \|^2 &\iff \min_{C \subseteq P:\ |C|=k} \left(kc - \frac{\| \sum_{V_i \in C} V_i \|^2}{k} \right) \\
&\iff \min_{C \subseteq P:\ |C|=k} \sum_{V_i \in C} \left(\|V_i\|^2 - \frac{\| \sum_{V_i \in C} V_i \|^2}{k} \right) \\
&\iff \min_{C \subseteq P:\ |C|=k} \sum_{V_i \in C} \| V_i - \frac{\sum_{V_i \in C} V_i}{k} \|^2 \\
&\iff \min_{C \subseteq P:\ |C|=k} \min_{p \in \Re^n} \sum_{V_i \in C} \|V_i - p\|^2
\end{aligned}$$

The first equivalence is straightforward. The second one is due to $\|V_i\|^2 = c$. The third one can be verified by simple calculation. The fourth one is because of Fact 1.2. Therefore:

Theorem 3.2 *BQP is equivalent to the* Densest-kPoints-Mean *problem.*

4 Approximation Algorithms for Densest k-Points

4.1 An 2-approximation algorithm

So far, we have casted Binary Quadratic Programming problem into the form of Densest-kPoints-Mean problem.

Here is a simple 2-approximation algorithm for Densest-kPoints problems ($\text{dist}(\cdot,\cdot)$ is a metric): given a set of n points P, for every point $p \in P$, let C_p be the k nearest points to p in P (including p itself); among the n choices of C_p's, we pick the one that minimize $\text{var}(C_p)$. It can be shown this is an 2-approximation algorithm for all the three versions of Densest-kPoints problems (depending on how $\text{var}(C_p)$ is defined): Densest-kPoints-Mean, Densest-kPoints-Center, and Densest-kPoints-Median.

Theorem 4.1 *This algorithm gives 2-approximation to* DENSEST-KPOINTS *problems.*

Proof. It is straightforward to prove the performance guarantee for the Center version and Median version. Following is the proof for the Means version (a bit trickier).

Suppose the optimal solution is $C^* = \{c_1^*, c_2^*, \ldots, c_k^*\}$, $c^* = \text{avg}(C^*) = \frac{\sum_{c_i^* \in C^*} c_i^*}{k}$ is the centroid of C^*. And suppose $C' = \{c_1', c_2', \ldots, c_k'\}$ is the solution found by our algorithm, and v_0 among P minimizes minimize the distance to c^*, $\|v_0 - c^*\|_2$. Note $\sum_{c_i^* \in C^*}(c_i^* - c^*) = 0$. It is clear that we have $\text{SOL} = \sum_{c_i' \in C'} \|c_i' - \text{avg}(C')\|_2^2 \leq \sum_{c_i^* \in C^*} \|c_i^* - v_0\|_2^2$. So,

$$
\begin{aligned}
\text{SOL} &\leq \sum_{c_i^* \in C^*} \|c_i^* - v_0\|_2^2 \\
&= \sum_{c_i^* \in C^*} \|(c_i^* - c^*) + (v_0 - c^*)\|_2^2 \\
&= \sum_{c_i^* \in C^*} (\|c_i^* - c^*\|_2^2 + 2(c_i^* - c^*)(v_0 - c^*) + \|v_0 - c^*\|_2^2) \\
&= \left(\sum_{c_i^* \in C^*} \|c_i^* - c^*\|_2^2 \right) + \left(2(v_0 - c^*)^T \sum_{c_i^* \in C^*} (c_i^* - c^*) \right) + k\|v_0 - c^*\|_2^2 \\
&= \left(\sum_{c_i^* \in C^*} \|c_i^* - c^*\|_2^2 \right) + k\|v_0 - c^*\|_2^2 \\
&\leq 2 \sum_{c_i^* \in C^*} \|c_i^* - c^*\|_2^2 = 2\text{OPT} \quad \text{(for } v_0 \text{ is the closed point to } c^* \text{ in } P\text{)}.
\end{aligned}
$$

Proofs for the Center version and Median version are similar and simpler. □

Like the algorithm in [18], we can further refine the output C' but iteratively taking the centroid $\text{avg}(C')$ and finding the nearest k points to $\text{avg}(C')$ in P to update C' until it converges. After the refinement process, we employ local search to interchange one element in C' and one in P/C' to see whether such change improves the solution. This step is expensive, so in our implementation we set the maximum number of iterations for local search as 10. This implementation is denoted by "One-Refine+Local Search".

We also implement another version "N-Refine+Local Search" by repeating the above refinement process and local search for every C_p besides the best one C'. Numerical results in Table 1 indicate that "N-Refine+Local Search" is slightly better than "One-Refine+Local Search", but consumes much more time.

4.2 $(1+\epsilon)$-approximation algorithms using exponential grids

One idea to improve the approximation ratio 2 is to use the vertices in a grid to approximate the center of the optimal solution to DENSEST-KPOINTS problems. The problem is there

Problem		One-Refine+Local Search		N-Refine+Local Search	
no.	k	Obj.	Time	Obj.	Time
M100_1	25	389.9279	0.1735	395.7495	0.1988
M100_1	50	1400.4021	0.5869	1410.4031	4.4459
MB100_1	25	334	0.2594	355	2.976
MB100_1	50	1095	0.4596	1111	2.316
M200_1	50	1491.2185	1.5505	1504.2743	35.1584
M200_1	100	5448.71	4.1615	5457.2287	43.7655
MB250_1	100	4399	6.2684	4435	89.952
MB250_1	150	9018	11.3736	9033	136.328
M500_1	100	4730	18.7091	4768	760.6783

Table 1: **Computational results for the maximum BQP problem**

might be too many vertices in the grid we need to check. We can extend the idea of [14] in high-dimension space to restrict the search space. Since this algorithm is extended from [14], we will only introduce the main ideas and state our main results below.

Given a set of n points P in $\Re^d$, we first use the algorithm introduced in Section 4.1 to get an 2-approximation C'. Suppose $\mathrm{SOL} = \mathrm{var}(C')$ and $\mathrm{OPT} = \mathrm{var}(C^*)$, where C^* is the optimal solution, then we have $\mathrm{SOL} \leq 2\mathrm{OPT}$. We know the radius of C^* (the distance from the center c^* of C^* to the farthest point in C^*) $\leq \mathrm{OPT}$ (or $\sqrt{\mathrm{OPT}}$ for the Mean version).

So, we first use a grid of size SOL (or $\sqrt{\mathrm{SOL}}$ for the Mean version) to cover the point set P. At the first glance, there is an unbounded number of squares in this grid we need to consider (if the scale of P is unbounded). But, we can observe that an optimal solution C^* may intersect with at most 3^d squares in this grid. Therefore, where are at most $n \cdot 3^d$ squares we need to consider.

For each of these squares we need to consider, we use a smaller grid of size $\frac{\epsilon \cdot \mathrm{SOL}}{2}$ (or $\frac{\epsilon \cdot \mathrm{SOL}}{2k}$ for the Median version, $\frac{\epsilon \cdot \sqrt{\mathrm{SOL}}}{\sqrt{2k}}$ for the Mean version) to cover it. We enumerate all the vertices in each small grid as the center of the solution, and output the best one.

Theorem 4.2 *Given a set of n points P, there are $(1+\epsilon)$-approximation algorithms with running time $O\left(\left(\frac{6}{\epsilon}\right)^d n^2\right)$ for* Densest-kPoints-Center, $O\left(\left(\frac{6k}{\epsilon}\right)^d n^2\right)$ *for* Densest-kPoints-Median, *and $O\left(\left(\frac{3\sqrt{2}k}{\epsilon}\right)^d n^2\right)$ for* Densest-kPoints-Mean.

Similar sampling techniques as in [14] might be applied to reduce the complexity from n^2 to n. However, when d is large, the grid-based algorithms discussed above does not scale well. In particular, when we transform the BQP problem into a Densest-kPoints-Mean problem, the d is equal to n. In the next subsection, we will show how the core-set techniques can be used to get faster algorithms, e.g. with running time $O(n^{1/\epsilon})$ or $O(2^{1/\epsilon}n)$.

4.3 Faster $(1+\epsilon)$-approximation algorithms using core-sets

The core-set techniques can be directly applied in DENSEST-KPOINTS problems to get faster algorithms. The main idea is: Consider an optimal solution C^* (of size k) for a set of n points P, if C^* has a core-set S, i.e. the center of S, c_S, is an approximation to the center of C^*, c_{C^*}, then we can first find c_S for some S, and find the nearest k points to c_S in P. It can be shown this is an $(1+\epsilon)$-approximation. The time complexity depends on how S is obtained (through enumeration or through sampling), and how fast c_S can be computed.

In the following part, we use $\text{cost}(C, x)$ to denote $\text{mean}(C, x)$, $\text{cen}(C, x)$, or $\text{med}(C, x)$. Recall the DENSEST-KPOINTS problem is: given a set of n points P in $\Re^d$, find $C \subseteq P$ of size k and $x \in \Re^d$ s.t. $\text{cost}(C, x)$ is minimized.

Fixing a point set S, let c_S be the point $x \in \Re^d$ that minimizes $\text{cost}(S, x)$.

4.3.1 Enumeration algorithm

Assumption 4.3 *For any $\epsilon < 1$, there exists a constant $m(\epsilon)$ s.t. for any point set X in $\Re^d$, there exists a subset (core-set) $S \subseteq X$ of size $m(\epsilon)$: (i) we can compute c_S in time $f(\epsilon)$; (ii) $\text{cost}(X, c_S) \leq (1+\epsilon)\text{cost}(X, c_X)$.*

Recall the core-set techniques surveyed in Section 2, the above assumption is valid the for Means/Center version of $\text{cost}(\cdot, \cdot)$. For the Median version, a slightly weaker assumption holds: we can find $g(\epsilon, |X|)$ candidates in $\Re^d$ based on S without knowing P, s.t. one of these candidates, denoted by c'_S, satisfies $\text{cost}(X, c_S) \leq (1+\epsilon)\text{cost}(X, c_X)$.

Given n points P and integer k, suppose C^* of size k is the optimal solution to the DENSEST-KPOINTS problem. From the above assumption, there exists a core-set S of size $m(\epsilon)$ for C^*. Therefore, our algorithm guess S by enumerating all $m(\epsilon)$-subsets of P, and for each possible subset S, we construct a solution by finding c_S and k nearest points to c_S in P. Finally, we pick the best solution. The running time is $O(n^{m(\epsilon)} \cdot f(\epsilon) \cdot nk)$.

For the Median version, we also need to try every candidate for a possible set S, so the running time is $O(n^{m(\epsilon)} \cdot g(\epsilon, k) \cdot nk)$.

In the Means/Center version, we have $m(\epsilon) = \frac{1}{\epsilon}$, and $f(\epsilon) = \text{poly}(\frac{1}{\epsilon})$, so the running time is $n^{O(1/\epsilon)}$. In the Median version, we have $m(\epsilon) = O(\frac{1}{\epsilon^4})$ and $g(\epsilon, k) = O(2^{O(1/\epsilon^4)} \log k)$, so the running time is $n^{O(1/\epsilon^4)}$.

Theorem 4.4 *There is an $(1+\epsilon)$-approximation algorithm with running time $n^{O(1/\epsilon)}$ for the* DENSEST-KPOINTS-MEAN/DENSEST-KPOINTS-CENTER *problem, or with running time $n^{O(1/\epsilon^4)}$ for the* DENSEST-KPOINTS-MEDIAN *problem.*

4.3.2 Sampling algorithm

Assumption 4.5 *For any $\epsilon < 1$, there exists a constant $m(\epsilon)$ s.t. for any point set X in $\Re^d$, from an uniformly random sample multiset (core-set) $S \subseteq X$ of size $m(\epsilon)$: (i) we can compute c_S in time $f(\epsilon)$; (ii) $\text{cost}(X, c_S) \leq (1+\epsilon)\text{cost}(X, c_X)$ holds with high probability.*

As is discussed in Section 2, this assumption is valid for the Means version. For the Median version, a slightly weaker assumption holds, but the algorithm and the analysis are similar. So we will focus on the Means version in the rest part.

Given n points P and integer k, suppose C^* of size k is the optimal solution to DENSEST-KPOINTS problem. If $k \geq \lambda n$, then, with high probability, a random sample multiset $S' \subseteq P$ of size $\frac{2}{\lambda \cdot \epsilon}$ contains at least $\frac{1}{\epsilon}$ points from C^* with high probability (from the Makov inequality). Let $S \subseteq S'$ denote these $\frac{1}{\epsilon}$ points (i.e. $S \subseteq C^*$ also). From Assumption 4.5 (note $m(\epsilon) = \frac{1}{\epsilon}$ for the Means version), with high probability, $\text{cost}(C^*, c_S) \leq (1+\epsilon)\text{cost}(C^*, c_{C^*})$. So the set of the k nearest points to c_S is an $(1+\epsilon)$-approximation for DENSEST-KPOINTS-MEAN with high probability. The only problem is that we do not know the $\frac{1}{\epsilon}$-subset $S \subseteq S'$ that satisfies $S \subseteq C^*$, so we enumerate all the $\frac{1}{\epsilon}$-subsets of S' to guess S.

Our algorithm works as follows. Sample a multiset S' of size $\frac{2}{\lambda \cdot \epsilon}$ from P. Enumerate all $\frac{1}{\epsilon}$-subsets S of S'. For each S, compute c_S and find the k nearest points to c_S in P—these k points form a candidate solution. Output the best among all candidate solutions.

Theorem 4.6 *If $k \geq \lambda n$ for some constant λ, there is a randomized $(1+\epsilon)$-approximation algorithm with running time $O(2^{O(1/\epsilon)}nk)$ for the* DENSEST-KPOINTS-MEAN *problem, or with running time $O(2^{O(1/\epsilon^4)}nk)$ for the* DENSEST-KPOINTS-MEDIAN *problem.*

Using similar analysis and the Chernoff bounds, we can relax the requirement on k a bit. It turns out that the problem is hard when k is "small" but not constant ($k \in \omega(1) \cap o(\log n)$).

Theorem 4.7 *If $k = \Omega(\frac{n \log m}{m})$, there is a randomized $(1 + \frac{1}{\log m})$-approximation algorithm with running time $O(2^{O(m)}nk)$ for the* DENSEST-KPOINTS-MEAN *problem, or with running time $O(2^{O(m^4)}nk)$ for the* DENSEST-KPOINTS-MEDIAN *problem.*

5 Conclusion and Future Work

In this report, we propose the DENSEST-KPOINTS-MEAN problems and approximation algorithms for it. The Binary Quadratic Programming problem is shown to be equivalent to DENSEST-KPOINTS-MEAN. We also did a survey on core-set techniques for clustering, and then use the core-set techniques to obtain $(1+\epsilon)$-approximation algorithms for the DENSEST-KPOINTS problems (which are faster than the grid-based $(1+\epsilon)$-approximation algorithms). In particular, we propose a PTAS for the BQP problem using core-sets.

Some interesting future work include:

Is there any faster algorithm for the DENSEST-KPOINTS problems when $k \in \omega(1) \cap o(\log n)$? Either when k is constant (then use the naive $O(n^k)$ algorithm) or when k is large (use the core-set techniques as in Section 4.3.2), the problem is easy. It is interesting whether there is faster $(1+\epsilon)$-approximation algorithm when k is in the middle.

It is also interesting whether the core-set techniques can be used in other clustering problems, like clustering uncertain data [7, 11] or coclustering problem [3], to obtain provably good results. We have found core-sets techniques can be applied in the MINSUMRADII

problem (minimize the sum of radii of clusters) to obtain $(1+\epsilon)$-approximation (the algorithm is identical to the one for k-Center clustering, introduced in Section 2.2).

References

[1] Marcel R. Ackermann and Johannes Blömer. Coresets and approximate clustering for bregman divergences. In *SODA*, pages 1088–1097, 2009.

[2] Marcel R. Ackermann, Johannes Blömer, and Christian Sohler. Clustering for metric and non-metric distance measures. In *SODA*, pages 799–808, 2008.

[3] Aris Anagnostopoulos, Anirban Dasgupta, and Ravi Kumar. Approximation algorithms for co-clustering. In *PODS*, pages 201–210, 2008.

[4] Mihai Badoiu and Kenneth L. Clarkson. Smaller core-sets for balls. In *SODA*, pages 801–802, 2003.

[5] Mihai Badoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *STOC*, pages 250–257, 2002.

[6] Ke Chen. On k-median clustering in high dimensions. In *SODA*, pages 1177–1185, 2006.

[7] Graham Cormode and Andrew McGregor. Approximation algorithms for clustering uncertain data. In *PODS*, pages 191–200, 2008.

[8] Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A ptas for k-means clustering based on weak coresets. In *Symposium on Computational Geometry*, pages 11–18, 2007.

[9] Gereon Frahling and Christian Sohler. A fast k-means implementation using coresets. In *Symposium on Computational Geometry*, pages 135–143, 2006.

[10] Ashish Goel, Piotr Indyk, and Kasturi R. Varadarajan. Reductions among high dimensional proximity problems. In *SODA*, pages 769–778, 2001.

[11] Sudipto Guha and Kamesh Munagala. Exceeding expectations and clustering uncertain data. In *PODS*, 2009, to appear.

[12] Sariel Har-Peled and Akash Kushal. Smaller coresets for k-median and k-means clustering. In *Symposium on Computational Geometry*, pages 126–134, 2005.

[13] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *STOC*, pages 291–300, 2004.

[14] Sariel Har-Peled and Soham Mazumdar. Fast algorithms for computing the smallest k-enclosing circle. *Algorithmica*, 41(3):147–157, 2005.

[15] Sariel Har-Peled and Kasturi R. Varadarajan. Projective clustering in high dimensions using core-sets. In *Symposium on Computational Geometry*, pages 312–318, 2002.

[16] Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering (extended abstract). In *Symposium on Computational Geometry*, pages 332–339, 1994.

[17] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. A simple linear time $(1+\epsilon)$-approximation algorithm for k-means clustering in any dimensions. In *FOCS*, pages 454–462, 2004.

[18] MacQueen and James B. Some methods for classification and analysis of multivariate observations. *Computer and Chemistry*, 4:257–272, 1967.

[19] P. M. Pardalos and J. Xue. The maximum clique problem. *Journal of Global Optimization*, 4:301–328, 1994.