A Simple Linear Time $(1+\varepsilon)$-Approximation Algorithm for k-Means Clustering in Any Dimensions

Amit Kumar
Dept. of Computer Science & Engg., IIT Delhi
New Delhi-110016, India
amitk@cse.iitd.ernet.in

Yogish Sabharwal
IBM India Research Lab
Block-I, IIT Delhi,
New Delhi-110016, India
ysabharwal@in.ibm.com

Sandeep Sen[1]
Dept. of Computer Science & Engg., IIT Delhi
New Delhi-110016, India
ssen@cse.iitd.ernet.in

Abstract

We present the first linear time $(1+\varepsilon)$-approximation algorithm for the k-means problem for fixed k and ε. Our algorithm runs in $O(nd)$ time, which is linear in the size of the input. Another feature of our algorithm is its simplicity – the only technique involved is random sampling.

1. Introduction

The problem of clustering a group of data items into similar groups is one of the most widely studied problems in computer science. Clustering has applications in a variety of areas, for example, data mining, information retrieval, image processing, and web search ([5, 7, 14, 9]). Given the wide range of applications, many different definitions of clustering exist in the literature ([8, 4]). Most of these definitions begin by defining a notion of distance between two data items and then try to form clusters so that data items with small distance between them get clustered together.

Often, clustering problems arise in a geometric setting, i.e., the data items are points in a high dimensional Euclidean space. In such settings, it is natural to define the distance between two points as the Euclidean distance between them. One of the most popular definitions of clustering is the *k-means clustering problem*. Given a set of points P, the k-means clustering problems seeks to find a set K of k centers, such that

$$\sum_{p \in P} d(p,K)^2$$

is minimized. Note that the points in K can be arbitrary points in the Euclidean space. Here $d(p,K)$ refers to the distance between p and the closest center in K. We can think of this as each point in P gets assigned to the closest center in K. The points that get assigned to the same center form a cluster. The k-means problem is NP-hard even for $k = 2$. Another popular definition of clustering is the k-median problem. This is defined in the same manner as the k-means problem except for the fact that the objective function is $\sum_{p \in P} d(p,K)$. Observe that the distance measure used in the definition of the k-means problem is not a metric. This might lead one to believe that solving the k-means problem is more difficult than the k-median problem. However, in this paper, we give strong evidence that this may not be the case.

A lot of research has been devoted to solving the k-means problem exactly (see [11] and the references therein). Even the best known algorithms for this problem take at least $\Omega(n^d)$ time. Recently, some work has been devoted to finding $(1+\varepsilon)$-approximation algorithms for the k-means problem, where ε can be an arbitrarily small constant. This has led to algorithms with much improved running time. Further, if we look at the applications of the k-means problem, they often involve mapping subjective features to points in the Euclidean space. Since there is an error inherent in this mapping, finding a $(1+\varepsilon)$-approximate solution does not lead to a deterioration in the solution for the actual application.

In this paper, we give the first truly linear time $(1+\varepsilon)$-approximation algorithm for the k-means problem. Treating k and ε as constants, our algorithm runs in $O(nd)$ time, which is linear in the size of the input. Another feature of our algorithm is its simplicity – the only technique involved is random sampling.

1.1. Related work

The fastest exact algorithm for the k-means clustering problem was proposed by Inaba et al. [11]. They observed that the number of Voronoi partitions of k points in $\Re^d$ is $O(n^{kd})$ and so the optimal k-means clustering could be de-

1 Author's present address: Dept of Computer Science and Engneering, IIT Kharagpur 721302.

termined exactly in time $O(n^{kd+1})$. They also proposed a randomized $(1+\varepsilon)$-approximation algorithm for the 2-means clustering problem with running time $O(n/\varepsilon^d)$.

Matousek [13] proposed a deterministic $(1+\varepsilon)$-approximation algorithm for the k-means problem with running time $O(n\varepsilon^{-2k^2 d}\log^k n)$. Badoiu et al. [3] proposed a $(1+\varepsilon)$-approximation algorithm for the k-median clustering problem with running time $O(2^{(k/\varepsilon)^{O(1)}} d^{O(1)} n\log^{O(k)} n)$. Their algorithm can be extended to get a $(1+\varepsilon)$-approximation algorithm for the k-means clustering problem with a similar running time. de la Vega et al. [6] proposed a $(1+\varepsilon)$-approximation algorithm for the k-means problem which works well for points in high dimensional points in high dimensions. The running time of this algorithm is $O(g(k,\varepsilon)n\log^k n)$ where $g(k,\varepsilon) = exp[(k^3/\varepsilon^8)(\ln(k/\varepsilon)\ln k]$. Recently, Har-Peled et al. [10] proposed a $(1+\varepsilon)$-approximation algorithm for the k-means clustering whose running time is $O(n + k^{k+2}\varepsilon^{-(2d+1)k}\log^{k+1} n\log^k \frac{1}{\varepsilon})$. Their algorithm is also fairly complicated and relies on several results in computational geometry that depend exponentially on the number of dimensions. So this is more suitable for low dimensions only.

There exist other definitions of clustering, for example, k-median clustering where the objective is to minimize the sum of the distances to the nearest center and k-center clustering, where the objective is to minimize the maximum distance (see [1, 2, 3, 10, 12] and references therein).

1.2. Our contributions

We present a linear time $(1+\varepsilon)$-approximation algorithm for the k-means problem. Treating k and ε as constants, the running time of our algorithm is better in comparison to the previously known algorithms for this problem. However, the algorithm due to Har-Peled and Mazumdar [10] deserves careful comparison. Note that their algorithm, though linear in n, is not linear in the input size of the problem, which is dn (for n points in d dimensions). Therefore, their algorithm is better only for low dimensions; for $d = \Omega(\log n)$, our algorithm is much faster. Even use of Johnson-Lindenstraus lemma will not make the running time comparable as it has its own overheads. Many recent algorithms rely on techniques like exponential grid or scaling that have high overheads. For instance, normalizing with respect to minimum distance between points may incur an extra $\Omega(n)$ cost per point depending on the computational model. In [3], the authors have used rounding techniques based on approximations of the optimal k-center value without specifying the cost incurred in the process. The techniques employed in our algorithm have no such hidden overheads.

The 2-means clustering problem has also generated enough research interest in the past. Our algorithm yields a $(1+\varepsilon)$-approximation algorithm for the 2-means clustering problem with constant probability in time $O(2^{(1/\varepsilon)^{O(1)}} dn)$. This is the first dimension independent (in the exponent) algorithm for this problem that runs in linear time.

The basic idea of our algorithm is very simple. We begin with the observation of Inaba et. al. [11] that given a set of points, their centroid can be very well approximated by sampling a constant number of points and finding the centroid of this sample. So if we knew the clusters formed by the optimal solution, we can get good approximations to the actual centers. Of course, we do not know this fact. However, if we sample $O(k)$ points, we know that we will get a constant number of points from the largest cluster. Thus, by trying all subsets of constant size from this sample, we can essentially sample points from the largest cluster. In this way, we can estimate the centers of large clusters. However, in order to sample from the smaller clusters, we need to prune points from the larger clusters. This pruning has to balance two facts – we would not like to remove points from the smaller clusters and yet we want to remove enough points from the larger clusters.

Our algorithm appears very similar in spirit to that of Badiou et al. [3]. In fact both these algorithms begin with the same premise of random sampling. However, in order to sample from the smaller clusters, their algorithm has to guess the sizes of the smaller clusters and the distances between clusters. This causes an $O(\log^k n)$ multiplicative factor in the running time of their algorithm. We completely avoid this extra factor by a much more careful pruning algorithm. Moreover this makes our algorithm considerably simpler.

2. Preliminaries

Let P be a set of n points in the Euclidean space $\Re^d$. Given a set of k points K, which we also denote as *centers*, define the *$k-$means cost of P with respect to K*, $\Delta(P,K)$, as

$$\Delta(P,K) = \sum_{p \in P} d(p,K)^2,$$

where $d(p,K)$ denotes the distance between p and the closest point to p in K. The $k-$means problem seeks to find a set[1] K of size k such that $\Delta(P,K)$ is minimized. Let $\Delta_k(P)$ denote the cost of the optimal solution to the $k-$means problem with respect to P.

1 In this paper we have addressed the unconstrained problem, where this set can consist of any k points in $\Re^d$.

If K happens to be a singleton set $\{y\}$, then we shall denote $\Delta(P, K)$ by $\Delta(P, y)$. Similar comments apply when P is a singleton set.

Definition 2.1. We say that the point set P is (k, ε)-*irreducible* if $\Delta_{k-1}(P) \geq (1+32\varepsilon)\Delta_k(P)$. Otherwise we say that the point set is (k, ε)-*reducible*.

Reducibility basically captures the fact that if instead of finding the optimal k-means solution, we find the optimal $(k-1)$-means solution, we will still be close to the former solution. We now look at some properties of the 1-means problem.

2.1. Properties of the 1-means problem

Definition 2.2. For a set of points P, define the *centroid*, $c(P)$, of P as the point $\frac{\sum_{p \in P} p}{|P|}$.

For any point $x \in \Re^d$, it is easy to check that

$$\Delta(P, x) = \Delta(P, c(P)) + |P| \cdot \Delta(c(P), x). \qquad (1)$$

From this we can make the following observation.

Fact 2.1. *Any optimal solution to the 1-means problem with respect to an input point set P chooses $c(P)$ as the center.*

We can also deduce an important property of any optimal solution to the k-means problem. Suppose we are given an optimal solution to the k-means problem with respect to the input P. Let $K = \{x_1, \ldots, x_k\}$ be the set of centers constructed by this solution. K produces a partitioning of the point set P into K clusters, namely, $P_1, \ldots, P_K$. P_i is the set of points for which the closest point in K is x_i. In other words, the clusters correspond to the points in the Voronoi regions in $\Re^d$ with respect to K. Now, Fact 2.1 implies that x_i must be the centroid of P_i for all i.

Since we will be interested in fast algorithms for computing good approximations to the k-means problem, we first consider the case $k = 1$. Inaba et. al. [11] showed that the centroid of a small random sample of points in P can be a good approximation to $c(P)$.

Lemma 2.2. [11] *Let T be a set of m points obtained by independently sampling m points uniformly at random from a point set P. Then, for any $\delta > 0$,*

$$\Delta(S, c(T)) \quad < \quad \left(1 + \frac{1}{\delta m}\right)\Delta_1(P)$$

holds with probability at least $1 - \delta$.

Therefore, if we choose m as $\frac{2}{\varepsilon}$, then with probability at least 1/2, we get a $(1+\varepsilon)$-approximation to $\Delta_1(P)$ by taking the center as the centroid of T. Thus, a constant size sample can quickly yield a good approximation to the optimal 1-means solution.

Suppose P' is a subset of P and we want to get a good approximation to the optimal 1-means for the point set P'. Following lemma 2.2, we would like to sample from P'. But the problem is that P' is not explicitly given to us. The following lemma states that if the size of P' is close to that of P, then we can sample a slightly larger set of points from P and hopefully this sample would contain enough random samples from P'. Let us define things more formally first. Let P be a set of points and P' be a subset of P such that $|P'| \geq \beta|P|$, where β is a constant between 0 and 1. Suppose we take a sample S of size $\frac{4}{\beta\varepsilon}$ from P. Now we consider all possible subsets of size $\frac{2}{\varepsilon}$ of S. For each of these subsets S', we compute its centroid $c(S')$, and consider this as a potential center for the 1-means problem instance on P'. In other words, we consider $\Delta(P', c(S'))$ for all such subsets S'. The following lemma shows that one of these subsets must give a close enough approximation to the optimal 1-means solution for P'.

Lemma 2.3. *(Superset Sampling Lemma) The following event happens with constant probability*

$$\min_{S':S' \subset S, |S'| = \frac{2}{\varepsilon}} \Delta(P', c(S')) \leq (1+\varepsilon)\Delta_1(P')$$

Proof. With constant probability, S contains at least $\frac{2}{\varepsilon}$ points from P'. The rest follows from Lemma 2.2. $\square$

We use the standard notation $\mathcal{B}(p, r)$ to denote the open ball of radius r around a point p.

We assume the input parameter ε for the approximation factor satisfies $0 < \varepsilon \leq 1$.

3. A linear time algorithm for 2-means clustering

Before considering the k-means problem, we consider the 2-means problem. This contains many of the ideas inherent in the more general algorithm. So it will make it easier to understand the more general algorithm.

Theorem 3.1. *Given a point set P of size n in $\Re^d$, there exists an algorithm which produces a $(1+\varepsilon)$-approximation to the optimal 2-means solution on the point set P with constant probability. Further, this algorithm runs in time $O(2^{(1/\varepsilon)^{O(1)}} dn)$.*

Proof. Let $\alpha = \varepsilon/64$. We can assume that P is $(2, \alpha)$-irreducible. Indeed suppose P is $(2, \alpha)$-reducible. Then $\Delta_1(P) \leq (1+\varepsilon/2)\Delta_2(P)$. We can get a solution to the 1-means problem for P by computing the centroid of P in $O(nd)$ time. The cost of this solution is at most $(1+\varepsilon/2)\Delta_2(P)$. Thus we have shown the theorem if P is $(2, \alpha)$-reducible.

Consider an optimal 2-means solution for P. Let c_1 and c_2 be the two centers in this solution. Let P_1 be the points which are closer to c_1 than c_2 and P_2 be the points closer to c_2 than c_1. So c_1 is the centroid of P_1 and c_2 that of P_2. Without loss of generality, assume that $|P_1| \geq |P_2|$.

Since $|P_1| \geq |P|/2$, Lemma 2.3 implies that if we sample a set S of size $O\left(\frac{1}{\varepsilon}\right)$ from P and look at the set of centroids of all subsets of S of size $\frac{2}{\varepsilon}$, then at least one of these centroids, call it c_1' has the property that $\Delta(P_1, c_1') \leq (1+\alpha)\Delta(P_1, c_1)$. Since our algorithm is going to cycle through all such subsets of S, we can assume that we have found such a point c_1'.

Let the distance between c_1 and c_2 be t, i.e., $d(c_1, c_2) = t$.

Lemma 3.2. $d(c_1, c_1') \leq t/4$.

Proof. Suppose $d(c_1, c_1') > t/4$. Equation (1) implies that

$$\Delta(P_1, c_1') - \Delta(P_1, c_1) = |P_1|\Delta(c_1, c_1') \geq \frac{t^2|P_1|}{16}.$$

But we also know that left hand side is at most $\alpha\Delta(P_1, c_1)$. Thus we get $t^2|P_1| \leq 16\alpha\Delta(P_1, c_1)$.

Applying Equation (1) once again, we see that

$$\Delta(P_1, c_2) = \Delta(P_1, c_1) + t^2|P_1| \leq (1+16\alpha)\Delta(P_1, c_1).$$

Therefore, $\Delta(P, c_2) \leq (1+16\alpha)\Delta(P_1, c_1) + \Delta(P_2, c_2) \leq (1+16\alpha)\Delta_2(P)$. This contradicts the fact that P is $(2, \alpha)$-irreducible. □

Now consider the ball $\mathcal{B}(c_1', t/4)$. The previous lemma implies that this ball is contained in the ball $\mathcal{B}(c_1, t/2)$ of radius $t/2$ centered at c_1. So $\mathcal{B}(c_1', t/4)$ is contained in P_1. Since we are looking for the point c_2, we can delete the points in this ball and hope that the resulting point set has a good fraction of points from P_2.

This is what we prove next. Let P_1' denote the point set $P_1 - \mathcal{B}(c_1', t/4)$. Let P' denote $P_1' \cup P_2$. As we noted above P_2 is a subset of P'.

Claim 3.3. $|P_2| \geq \alpha|P_1'|$

Proof. Suppose not, i.e., $|P_2| \leq \alpha|P_1'|$. Notice that

$$\Delta(P_1, c_1') \geq \Delta(P_1', c_1') \geq \frac{t^2|P_1'|}{16}.$$

Since $\Delta(P_1, c_1') \leq (1+\alpha)\Delta(P_1, c_1)$, it follows that

$$t^2|P_1'| \leq 16(1+\alpha)\Delta(P_1, c_1) \tag{2}$$

So,

$$\begin{aligned}
\Delta(P, c_1) &= \Delta(P_1, c_1) + \Delta(P_2, c_1) \\
&= \Delta(P_1, c_1) + \Delta(P_2, c_2) + t^2|P_2| \\
&\leq \Delta(P_1, c_1) + \Delta(P_2, c_2) \\
&\quad +16\alpha(1+\alpha)\Delta(P_1, c_1) \\
&\leq (1+32\alpha)\Delta(P_1, c_1) + \Delta(P_2, c_2) \\
&\leq (1+32\alpha)\Delta_2(P),
\end{aligned}$$

where the second equation follows from (1), while third inequality follows from (2) and the fact $|P_2| \leq \alpha|P_1'|$. But this contradicts the fact that P is $(2, \alpha)$-irreducible. This proves the claim. □

The above claim combined with Lemma 2.2 implies that if we sample $O\left(\frac{1}{\alpha^2}\right)$ points from P', and consider the centroids of all subsets of size $\frac{2}{\alpha}$ in this sample, then with constant probability we shall get a point c_2' for which $\Delta(P_2, c_2') \leq (1+\alpha)\Delta(P_2, c_2)$. Thus, we get the centers c_1' and c_2' which satisfy the requirements of our lemma.

The only problem is that we do not know the value of the parameter t. We will somehow need to guess this value and yet maintain the fact that our algorithm takes only linear amount of time.

We can assume that we have found c_1' (this does not require any assumption on t). Now we need to sample from P' (recall that P' is the set of points obtained by removing the points in P distant at most $t/4$ from c_1'). Suppose we know the parameter i such that $\frac{n}{2^i} \leq |P'| \leq \frac{n}{2^{i-1}}$.

Consider the points of P in descending order of distance from c_1'. Let Q_i' be the first $\frac{n}{2^{i-1}}$ points in this sequence. Notice that P' is a subset of Q_i' and $|P'| \geq |Q_i'|/2$. Also we can find Q_i' in linear time (because we can locate the point at position $\frac{n}{2^{i-1}}$ in linear time). Since $|P_2| \geq \alpha|P'|$, we see that $|P_2| \geq \alpha|Q_i'|/2$. Thus, Lemma 2.2 implies that it is enough to sample $O\left(\frac{1}{\alpha^2}\right)$ points from Q_i' to locate c_2' (with constant probability of course).

But the problem with this scheme is that we do not know the value i. One option is try all possible values of i, which will imply a running time of $O(n \log n)$ (treating the terms involving α and d as constant). Also note that we cannot use approximate range searching because preprocessing takes $O(n\text{log}n)$ time.

We somehow need to combine the sampling and the idea of guessing the value of i. Our algorithm proceeds as follows. It tries values of i in the order $0, 1, 2, \ldots$. In iteration i, we find the set of points Q_i'. Note that Q_{i+1}' is a subset of Q_i'. In fact Q_{i+1}' is the half of Q_i' which is farther from c_1'. So in iteration $(i+1)$, we can begin from the set of points Q_i' (instead of P'). We can find the candidate point c_2' by sampling from Q_{i+1}'. Thus we can find Q_{i+1}' in time linear in $|Q_{i+1}'|$ only.

Further in iteration i, we also maintain the sum $\Delta(P - Q_i', c_1')$. Since $\Delta(P - Q_{i+1}', c_1') = \Delta(P - Q_i', c_1') + \Delta(Q_i' - Q_{i+1}', c_1')$, we can compute $\Delta(P - Q_{i+1}', c_1')$ in iteration $i+1$ in time linear in Q_{i+1}'. This is needed because when we find a candidate c_2' in iteration $i+1$, we need to compute the 2-means solution when all points in $P - Q_i'$ are assigned to c_1' and the points in Q_i' are assigned to the nearer of c_1' and c_2'. We can do this in time linear in $|Q_{i+1}'|$ if we maintain the quantities $\Delta(P - Q_i', c_1')$ for all i.

Thus, we see that iteration i takes time linear in $|Q_i'|$. Since $|Q_i'|$'s decrease by a factor of 2, the overall running time for a given value of c'_1 is $O(2^{(1/\alpha)^{O(1)}}dn)$. Since the number of possible candidates for c'_1 is $O(2^{(1/\alpha)^{O(1)}})$, the running time is as stated.

Claim 3.4. *The cost, Δ, reported by the algorithm satisfies $\Delta_2(P) \leq \Delta \leq (1+\alpha)\Delta_2(P)$.*

Proof. $\Delta_2(P) \leq \Delta$ is obvious as we are associating each point with one of the 2 centers being reported and accumulating the corresponding cost. Now, consider the case when we have the candidate center set where each center is a $(1+\alpha)$-approximate centroid of it's respective cluster. As we are associating each point to the approximate centroid of the corresponding cluster or a center closer than it, it follows that $\Delta \leq (1+\alpha)\Delta_2(P)$. If we report the minimum cost clustering, $\mathcal{C}$, then since the actual cost of the clustering (due to the corresponding Voronoi partitioning) can only be better than the cost that we report (because we associate some points with approximate centroids of corresponding cluster rather than the closest center), we have $\Delta(\mathcal{C}) \leq \Delta \leq (1+\alpha)\Delta_2(P)$. □

This proves the theorem. □

4. A linear time algorithm for k-means clustering

We now present the general k-means algorithm. We first present a brief outline of the algorithm.

4.1. Outline

Our algorithm begins on the same lines as the 2-means algorithm. Again, we can assume that the solution is irreducible, i.e., removing one of the centers does not create a solution which has cost within a small factor of the optimal solution. Consider an optimal solution which has centers $c_1, \ldots, c_k$ and which correspondingly partitions the point set P into clusters $P_1, \ldots, P_k$. Assume that $|P_1| \geq \cdots \geq |P_k|$. Our goal again will be to find approximations $c'_1, \ldots, c'_k$ to $c_1, \ldots, c_k$ respectively.

Suppose we have found centers $c'_1, \ldots, c'_i$. Suppose t is the distance between the closest pair of centroids $\{c_1, \ldots, c_i\}$ and $\{c_{i+1}, \ldots, c_k\}$. As in the case of $k = 2$, we can show that the points at distant at most $t/4$ from $\{c'_1, \ldots, c'_i\}$ get assigned to $c_1, \ldots, c_i$ by the optimal solution. So, we can delete these points. Now we can show that among the remaining points, the size of P_{i+1} is significant. Therefore, we can use random sampling to obtain a center c'_{i+1} which is a pretty good estimate of c_{i+1}. Of course we do not know the value of t, and so a naive implementation of this idea gives an $O(n(\log n)^k)$ time algorithm.

Algorithm k**-means**(P, k, ε)
Inputs : Point set P,
Number of clusters k,
Approximation ratio ε.
Output : k-means clustering of P.

1. For $i = 1$ to k do
   Obtain the clustering
   Irred-k**-means**$(P, i, i, \phi, \varepsilon/64k, 0)$.
2. Return the clustering which has minimum cost.

Figure 1. The k-means Algorithm

So far the algorithm looks very similar to the $k = 2$ case. But now we want to modify it to a linear time algorithm. This is where the algorithm gets more involved. As mentioned above, we can not guess the parameter t. So we try to guess the size of the point set obtained by removing the balls of radius $t/4$ around $\{c_1, \ldots, c_i\}$. So we work with the remaining point set with the hope that the time taken for this remaining point set will also be small and so the overall time will be linear. Although similar in spirit to the $k = 2$ case, we still need to prove some more details in this case. Now, we describe the actual k-means algorithm.

4.2. The algorithm

The algorithm is described in Figures 1 and 2. Figure 1 is the main algorithm. The inputs are the point set P, k and an approximation factor ε. Let α denote $\varepsilon/64k$. The algorithm k**-means**(P, k, ε) tries to find the highest i such that P is (i, α)-irreducible. Essentially we are saying that it is enough to find i centers only. Since we do not know this value of i, the algorithm tries all possible values of i.

We now describe the algorithm **Irred-**k**-means**$(Q, m, k, C, \alpha, \text{Sum})$. We have found a set C of $k - m$ centers already. The points in $P - Q$ have been assigned to C. We need to assign the remaining points in Q. The case $m = 0$ is clear. In step 2, we try to find a new center by the random sampling method. This will work provided a good fraction of the points in Q do not get assigned to C. If this is not the case then in step 3, we assign half of the points in Q to C and call the algorithm recursively with this reduced point set. For the base case, when $|C| = 0$, as P_1 is the largest cluster, we require to sample only $O(k/\alpha)$ points. This is tackled in Step 2. Step 3 is not performed in this case, as there are no centers.

Algorithm Irred-k-means$(Q, m, k, C, \alpha, \text{Sum})$
Inputs Q: Remaining point set
m: number of cluster centers yet to be found
k: total number of clusters
C: set of $k-m$ cluster centers found so far
α: approximation factor
Sum: the cost of assigning points in $P-Q$ to the centers in C
Output The clustering of the points in Q in k clusters.

1. If $m = 0$
   Assign the points in Q to the nearest centers in C.
   Sum = Sum + The clustering cost of Q.
   Return the clustering.
2. (a) Sample a set S of size $O\left(\frac{k}{\alpha^2}\right)$ from Q.
   (b) For each set subset S' of S of size $O\left(\frac{1}{\alpha}\right)$ do
   Compute the centroid c of S'.
   Obtain the clustering
   Irred-k-means$(Q, m-1, k, C \cup \{c\}, \alpha, \text{Sum})$.
3. (a) Consider the points in Q in ascending order of distance from C.
   (b) Let U be the first $|Q|/2$ points in this sequence.
   (c) Assign the points in U to the nearest centers in C.
   (d) Sum = Sum + The clustering cost of U.
   (e) Compute the clustering
   Irred-k-means$(Q-U, m, k, C, \alpha, \text{Sum})$.
4. Return the clustering which has minimum cost.

Figure 2. The irreducible k-means algorithm

4.3. Correctness and running time

Theorem 4.1. *Suppose a point set P is (k, α)-irreducible. Then the algorithm* **Irred-k-means**$(P, k, k, \emptyset, \alpha, 0)$ *returns a k-means solution on input P of cost at most $(1+\alpha)\Delta_k(P)$ with probability γ^k, where γ is a constant, independent of k and ε.*

Proof. Consider an optimal k-means solution on input P. Let the centers be $\{c_1, \ldots, c_k\}$ and let these partition the point set P into clusters $P_1, \ldots, P_k$ respectively. The only source of randomization in our algorithm is the invocations to the superset sampling lemma (Lemma 2.3). Recall that the desired event in the superset sampling lemma happens with constant probability. For ease of exposition, we shall assume that this desired event in fact always happens when we invoke this lemma. At the end of this proof, we will compute the actual probability with which our algorithm succeeds. Thus, unless otherwise stated, we assume that the desired event in the superset sampling lemma always happens.

Observe that when we call **Irred-k-means** with input $(P, k, k, \emptyset, \alpha, 0)$, it gets called recursively again several times (although with different parameters). Let $\mathcal{C}$ be the set of all calls to **Irred-k-means** when we start it with input $(P, k, k, \emptyset, \alpha, 0)$. Let $\mathcal{C}_i$ be those calls in $\mathcal{C}$ in which the parameter C (i.e., the set of centers already found) has size i.

For all values of i, our algorithm shall maintain the following invariant :

> **Invariant :** The set $\mathcal{C}_i$ contains a call in which the list of parameters $(Q, m, k, C, \alpha, \text{Sum})$ has the following properties :
>
> (1) Let $C = \{c'_1, \ldots, c'_i\}$. Then for $j = 1, \ldots, i$, $\Delta(P_j, c'_j) \leq (1+\alpha)\Delta(P_j, c_j)$.
>
> (2) The set $P - Q$ is a subset of $P_1 \cup \cdots \cup P_i$.

Clearly, if we show that the invariant holds for $i = k$, then we are done. It holds trivially for $i = 0$. Suppose the invariant holds for some fixed i. We shall show that the invariant holds for $(i+1)$ as well.

Since the invariant holds for i, there exist parameter lists in $\mathcal{C}_i$ which satisfy the invariant mentioned above. Among such parameter lists, choose a list $(Q, m, k, C, \alpha, \text{Sum})$ for which $|Q|$ is smallest. Consider the closest pair of centers between the sets $\{c_1, \ldots, c_i\}$ and $\{c_{i+1}, \ldots, c_k\}$ – let these centers be c_r and c_l respectively. Let $t = d(c_r, c_l)$.

Lemma 4.2. *Let S be the set of points $\mathcal{B}(c'_1, t/4) \cup \cdots \cup \mathcal{B}(c'_i, t/4)$, i.e., the points which are distant at most $t/4$ from $\{c'_1, \ldots, c'_i\}$. Then S is contained in $P_1 \cup \cdots \cup P_i$. Further, $P - S$ contains at most $|P_{i+1}|/\alpha$ points of $P_1 \cup \cdots \cup P_i$.*

Proof. Suppose for the sake of contradiction that P_j contains a point from S, $j > i$. Say P_j contains a point x of $\mathcal{B}(c'_q, t/4)$, $q \leq i$.

Claim 4.3. $d(c_q, c'_q) \geq t/4$.

Proof. Suppose not. Then distance between $d(c_q, x) < t/2$. Note that x is assigned to the center closest to it. So, $d(c_j, x) \leq d(c_q, x) < t/2$. So, $d(c_j, c_q) < t$, which is a contradiction. □

We know that $\Delta(P_q, c'_q) \leq (1+\alpha)\Delta(P_q, c_q)$. But we know from equation (1) that $\Delta(P_q, c'_q) = \Delta(P_q, c_q) + |P_q| d(c_q, c'_q)^2$. Thus we get

$$|P_q| d(c_q, c'_q)^2 \leq \alpha \Delta(P_q, c_q). \tag{3}$$

Now observe that $d(c_q, c_j) \leq d(c_q, c'_q) + d(c'_q, x) + d(x, c_j)$. Also, $d(x, c_j) \leq d(x, c_q) \leq d(x, c'_q) + d(c_q, c'_q)$. Thus, we get $d(c_q, c_j) \leq 2d(c'_q, x) + 2d(c_q, c'_q)$. From the claim above, we get $d(c_q, c_j) \leq 4d(c_q, c'_q)$. But suppose we assign all the points in P_q to c_j. Let us compute the cost of doing this.

$$\begin{aligned}\Delta(P_q, c_j) &= \Delta(P_q, c_q) + |P_q| d(c_q, c_j)^2 \\ &\leq \Delta(P_q, c_q) + 16|P_q| d(c_q, c'_q)^2 \\ &\leq (1 + 16\alpha)\Delta(P_q, c_q)\end{aligned}$$

where the last inequality follows from the equation (3). But this violates the condition that P is (k, α)-irreducible. So, S is contained in $P_1 \cup \cdots \cup P_i$.

Recall that the closest pair of centers between $\{c_1, \ldots, c_i\}$ and $\{c_{i+1}, \ldots, c_k\}$ are c_r and c_l respectively. Suppose $P - S$ contains more than $|P_l|/\alpha$ points of $P_1 \cup \cdots \cup P_i$. In that case, these points are assigned to centers at distance at least $t/4$. It follows that the cost of the optimal solution $\Delta_k(P)$ is at least $\frac{t^2|P_l|}{16\alpha}$. In other words, $t^2|P_l| \leq 16\alpha\Delta_k(P)$. But then if we assign all the points in P_l to c_r, the cost increases by at most $16\alpha\Delta_k(P)$, which implies that P is (k, α)-reducible, a contradiction. This proves the lemma. □

Recall that we are looking at the parameter list $(Q, m, k, C, \alpha, \text{Sum})$ which satisfies the invariant for i. As in the Lemma above, let S denote the point set $\mathcal{B}(c'_1, t/4) \cup \cdots \cup \mathcal{B}(c'_i, t/4)$. Let P' denote $P - S$. We know that $P_{i+1} \cup \cdots \cup P_k$ is contained in $P' \cap Q$.

Claim 4.4. $|P_{i+1}| \geq \frac{\alpha}{k}|P'|$.

Proof. By the Lemma above, there are at most $|P_{i+1}|/\alpha$ elements of $P_1 \cup \ldots \cup P_i$ in P'. So $|P'| \leq |P_{i+1}|/\alpha + |P_{i+1}| + \ldots + |P_k| \leq |P_{i+1}|/\alpha + k|P_{i+1}| \leq \frac{k}{\alpha}|P_{i+1}|$. □

It follows that $|P_{i+1}| \geq \frac{\alpha}{k}|P' \cap Q|$. So, if we knew P', then using Lemma 2.3, we can get a point c'_{i+1} which is a close approximation to c_{i+1} by sampling $O(k/\alpha^2)$ points from $P' \cap Q$. But of course we do not know P'.

Lemma 4.5. $|P' \cap Q| \geq |Q|/2$.

Proof. Suppose not, i.e., $|P' \cap Q| \leq |Q|/2$.

Claim 4.6. *Consider the points in Q sorted in ascending order of the distance from C. Let U be the first $|Q|/2$ points in this order. Then U does not contain a point of $P' \cap Q$.*

Proof. Suppose $x \in P'$ and $y \in P - P'$. Then we claim that y is closer to C than x is. Indeed, by definition of P', there is a center $c \in \{c_1, \ldots, c_i\}$ such that $d(c, y) \leq t/4$. If x were closer to C than y is, then there is a center in $\{c_1, \ldots, c_i\}$ whose distance from x is at most $t/4$. But then $x \in P - P'$, a contradiction. □

So, if U is as defined in the claim above, then $P' \cap Q$ is a subset of $Q - U$. Since $P_{i+1} \cup \cdots \cup P_k$ is contained in $P' \cap Q$ (because of Lemma 4.2 and the fact that Q is in the parameter list which satisfies the invariant for i), it follows that $P_{i+1} \cup \cdots \cup P_k$ is a subset of $Q - U$. Thus, the parameter list $(Q - U, C, k, m, \alpha, \text{Sum})$ which is formed in Step(e) of the algorithm satisfies the invariant for i as well, i.e., it is in $\mathcal{C}_i$. But this violates the fact that $(Q, C, k, m, \alpha, \text{Sum})$ was the parameter list satisfying the invariant for i in $\mathcal{C}_i$ for which $|Q|$ is smallest. This proves the lemma. □

The lemma above implies that $|P' \cap Q| \geq |Q|/2$. Combined with Claim 4.4, we get $|P_{i+1}| \geq \frac{\alpha|Q|}{4k}$. The superset sampling lemma combined with the claim above imply that by sampling $O(k/\alpha^2)$ points from Q, we shall get a point c'_{i+1} such that $\Delta(P_{i+1}, c'_{i+1}) \leq (1+\alpha)\Delta(P_{i+1}, c_{i+1})$. This is the case handled by the step 2(b) in the algorithm **Irred-k-means**. In this case the algorithm is called again with parameters $(Q, m-1, k, C \cup \{c'_{i+1}\}, \alpha, \text{Sum})$. It is easy to see now that this parameter list satisfies the invariant for $i + 1$. Thus we have shown that the invariant holds for all values of i.

As we mentioned earlier, a parameter list $(Q, m, k, C, \alpha, \text{Sum})$ which satisfies the invariant for $i = k$ has the desired centers in C.

Claim 4.7. *The cost, Δ, reported by the algorithm satisfies* $\Delta_k(P) \leq \Delta \leq (1 + \alpha)\Delta_k(P)$.

Proof. $\Delta_k(P) \leq \Delta$ is obvious as we are associating each point with one of the k centers being reported and accumulating the corresponding cost. Now, consider the case when we have the candidate center set where each center is a $(1 + \alpha)$-approximate centroid of it's respective cluster. As we are associating each point to the approximate centroid of the corresponding cluster or a center closer than it, it follows that $\Delta \leq (1 + \alpha)\Delta_k(P)$. If we report the minimum cost clustering, $\mathcal{C}$, then since the actual cost of the clustering (due to the corresponding Voronoi partitioning) can only be better than the cost that we report (because we associate some points with approximate centroids of corresponding cluster rather than the closest center), we have $\Delta(\mathcal{C}) \leq \Delta \leq (1 + \alpha)\Delta_k(P)$.

□

This proves the correctness of our algorithm. We just need to calculate the probability with which the algorithm is called with such a parameter list.

Note that the only source of randomness in **Irred-k-means** is in the Step 2(a). The sampling gives the desired result with constant probability (according to Lemma 2.3). Further each time we execute Step 2, we decrease m by 1. So, in any sequence of successive recursive calls, there can be at most k invocations of Step 2. Now, we have just shown that there is a parameter list in $\mathcal{C}_k$ for which C contains a set of centers close to the optimal clusters. Let us look at the sequence of recursive calls which have resulted in this parameter list. In these sequence of calls, as we mentioned above, there are k invocations of the random sampling. Each of these work correctly with constant probability. Therefore, the probability that we actually see this parameter list during the execution of this algorithm is γ^k for some constant γ. $\square$

Now we establish the running time of our algorithm.

Theorem 4.8. *The algorithm* **Irred-k-means** *when called with parameters* $(P, k, k, \emptyset, \alpha, 0)$ *runs in time* $O(2^{(k/\alpha)^{O(1)}} dn)$*, where* $n = |P|$.

Proof. Let $T(n, m)$ be the running time of our algorithm on input $(Q, m, k, C, \alpha, \mathrm{Sum})$ where $n = |Q|$. Then in Step 2(b), we have $u(k, \alpha)$ subsets of the sample, where $u(k, \alpha) = O(2^{(k/\alpha)^{O(1)}})$. Computation of a centroid of any set S' in Step 2(b) takes $O(d)$ time. Steps 3(a)-(d) take $O(nd)$ time. Therefore we get the recurrence

$$\begin{aligned} T(n, m) &= O(u(k, \alpha))T(n, m-1) + T(n/2, m) \\ &\quad + O((n + u(k, \alpha))d). \end{aligned}$$

It is not difficult to show from this that $T(n, k)$ is $O(2^{(k/\alpha)^{O(1)}} dn)$. $\square$

We can now state our main theorem.

Theorem 4.9. *A* $(1+\varepsilon)$*-approximate* k*-means clustering of a point set* P *in* $\Re^d$ *can be found in time* $O(2^{(k/\varepsilon)^{O(1)}} dn)$*, with constant probability.*

Proof. We can run the algorithm **Irred-k-means** c^k times for some constant c to ensure that it yields the desired result with constant probability. This still keeps the running time $O(2^{(k/\alpha)^{O(1)}} dn)$. So let us assume this algorithm gives the desired solution with constant probability.

Notice that the running time of our main algorithm in Figure 1 is also $O(2^{(k/\alpha)^{O(1)}} dn)$. We just have to show that it is correct.

Let i be the highest index for which P is (i, α)-irreducible. So, it follows that

$$\Delta_i(P) \leq (1+32\alpha)\Delta_{i+1}(P) \leq \cdots \leq (1+32\alpha)^{k-i}\Delta_k(P).$$

Further, we know that the algorithm **Irred-k-means** on input $(P, i, i, \emptyset, \alpha, 0)$ yields a set of i centers C for which $\Delta(P, C) \leq (1 + \alpha)\Delta_i(P)$. Therefore, we get a solution of cost at most $(1 + 32\alpha)^k \Delta_k(P)$. Note that $(1 + 32\alpha)^k = (1 + \varepsilon/2k)^k \leq 1 + \varepsilon$. This proves the theorem. $\square$

5. Concluding remarks

Our algorithm is very well suited for handling outliers - in fact, it becomes simpler.

Using the notion of balanced clusters in conjunction with Lemma 2.2, by eliminating at most $(1+\mu)\gamma|P|$ outliers, we can approximate the cost of the optimal k-means clustering with at most $\gamma|P|$ outliers.

An interesting direction for further research is to extend our methods for other clustering problems. Also, it is an open problem to get a polynomial time $(1 + \varepsilon)$-approximation algorithm for the k-means clustering problem when n, k and d are not constants.

References

[1] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for euclidean k-medians and related problems. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 106–113. ACM Press, 1998.

[2] V. Arya, N. Garg, R. Khandekar, K. Munagala, and V. Pandit. Local search heuristic for k-median and facility location problems. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 21–29. ACM Press, 2001.

[3] M. Bādoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 250–257. ACM Press, 2002.

[4] M. Bern and D. Eppstein. Approximation algorithms for geometric problems. pages 296–345, 1997.

[5] A. Broder, S. Glassman, M. Manasse, and G. Zweig. Syntactic clustering of the web. In *Proc. of 6th International World Wide Web Conference*, 1997.

[6] W. F. de la Vega, M. Karpinski, C. Kenyon, and Y. Rabani. Approximation schemes for clustering problems. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 50–58. ACM Press, 2003.

[7] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis, 1990.

[8] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[9] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and effective querying by image content. *J. Intell. Inf. Syst.*, 3(3-4):231–262, 1994.

[10] S. Har-Peled and S. Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 291–300. ACM Press, 2004.

[11] M. Inaba, N. Katoh, and H. Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering: (extended abstract). In *Proceedings of the tenth annual symposium on Computational geometry*, pages 332–339. ACM Press, 1994.

[12] S. Kolliopoulos and S. Rao. A nearly linear time approximation scheme for the euclidean k-medians problem. In *Proceedings of the 7th European Symposium on Algorithms*, pages 326–371, 1999.

[13] J. Matousek. On approximate geometric k-clustering. In *Discrete and Computational Geometry*, pages 61–84, 2000.

[14] M. J. Swain and D. H. Ballard. Color indexing. In *International Journal of Computer Vision*, pages 11–32, 1991.