S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11

cgt780

[DTD5] P.1(1-14)
by:violeta p. 1

Available online at www.sciencedirect.com
SCIENCE DIRECT®

Computational Geometry ●●● (●●●●) ●●●–●●●

Computational Geometry
Theory and Applications

www.elsevier.com/locate/comgeo

A linear time algorithm for approximate 2-means clustering

Yogish Sabharwal [a,*], Sandeep Sen [b,1]

[a] *IBM India Research Lab, Block-I, IIT Delhi, Hauz Khas, New Delhi-110016, India*
[b] *Department of Computer Science & Engg, Indian Institute of Technology, New Delhi-110016, India*

Received 26 March 2004; received in revised form 13 January 2005; accepted 14 January 2005

Communicated by P. Agarwal

Abstract

Matousek [Discrete Comput. Geom. 24 (1) (2000) 61–84] designed an $O(n \log n)$ deterministic algorithm for the approximate 2-means clustering problem for points in fixed dimensional Euclidean space which had left open the possibility of a linear time algorithm. In this paper, we present a simple randomized algorithm to determine an approximate 2-means clustering of a given set of points in fixed dimensional Euclidean space, with constant probability, in linear time.

We first approximate the mean of the larger cluster using random sampling. We then show that the problem can be reduced to a set of lines, on which it can be solved by carefully pruning away points of the larger cluster and randomly sampling on the remaining points to obtain an approximate to the mean of the smaller cluster.

1. Introduction

Clustering is the grouping of similar objects and a clustering of a set is a partition of its elements that minimizes some measure of dissimilarity. Clustering has applications in a variety of areas, for example, data mining, information retrieval, image processing, and web search [1,2,4,11].

* Corresponding author.
E-mail addresses: ysabharwal@in.ibm.com (Y. Sabharwal), ssen@cse.iitd.ernet.in (S. Sen).
[1] Author's present address: Department of Computer Science & Engg, Indian Institute of Technology Kharagpur 721302, India.

0925-7721/$ – see front matter © 2005 Published by Elsevier B.V.
doi:10.1016/j.comgeo.2005.01.003

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.2(1-14)
by:violeta p. 2

A popular clustering method for points in Euclidean space is k-means clustering. For completeness, we begin with some formal definitions.

Definition 1.1 (*k-means clustering*). Given a set P of n data points in d-dimensional Euclidean space, $\Re^d$, and an integer k, the problem is to determine a set, K, of k points in $\Re^d$, called centers, that minimize the mean squared Euclidean distance from each data point to its nearest center. This measure is often called the squared error distortion (Δ).

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

where $d(p, K) =$ Euclidean distance of p from its closest point in K.

The clusters $S_1, S_2, \ldots, S_k$ of P are derived from the set $K = \{x_1, x_2, \ldots, x_k\}$. The cluster S_i is derived from the point x_i by taking those points of P for which x_i is the closest point in K. Thus the points are partitioned by the Voronoi diagram of the set of centers.

Moreover, x_i is the centroid of set S_i, i.e.,

$$x_i = C(S_i) = \frac{1}{|S_i|} \sum_{s \in S_i} s.$$

Clustering based on k-means is closely related to a number of other clustering problems. These include the k-medians, in which the objective is to minimize the sum of the distances to the nearest center, and the k-center problem, in which the objective is to minimize the maximum distance.

Polynomial time approximation algorithms for clustering problems are very important considering that the exact solutions are often intractable [5,10]. The approximation version of the k-means problem is to determine a solution that is within a constant factor of the optimal solution.

Definition 1.2 (*$(1+\varepsilon)$-approximate k-means clustering*). Given a set P of n data points in d-dimensional Euclidean space, $\Re^d$, an integer k, and an approximation factor $\varepsilon > 0$, the problem is to determine a set of k points in $\Re^d$, so that the squared error distortion, Δ is within a factor of $(1+\varepsilon)$ of the optimal squared error distortion Δ_O, i.e.,

$$\Delta \leqslant (1+\varepsilon)\Delta_O.$$

The 2-means clustering problem is the special case of the k-means clustering problem where $k = 2$. Even the 2-means clustering problem is NP-hard when d is not fixed [3].

We use Δ to denote the k-means cost of a point set with respect to a given point, i.e.,

$$\Delta(S, y) = Cost_{k\text{-means}}(S, y) = \sum_{x \in S} \|x - y\|^2.$$

When the given point y is the centroid ($y = C(S) = \frac{1}{|S|} \sum_{x \in S} x$) of the given point set S, then we simply denote it as $\Delta(S)$, i.e.,

$$\Delta(S) = \sum_{x \in S} \big\|x - C(S)\big\|^2.$$

When the set S is a singleton $\{x\}$, then we simply denote it as $\Delta(x, y)$,i.e.,

$$\Delta(x, y) = \|x - y\|^2.$$

This is simply the square of the Euclidean metric.

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.3 (1-14)
by:violeta p. 3

1.1. Our results

We present a randomized algorithm that determines the approximate $(1+\varepsilon)$ 2-means clustering of a given point set in linear time, with constant probability, for any dimension.

It is interesting that even the 2-means clustering problem has eluded efficient and simple algorithms for a long time. Inaba et al. [7] gave an algorithm to compute the $(1+\frac{1}{\Omega(n\log n)})$ approximate clustering on the plane in $O(n^{5/3}\log^2 n)$ time with probability $(1-\frac{1}{\Omega(n\log n)})$. For higher dimensions, the running time of their algorithm increases exponentially in d. Recently, Matousek [9] gave an $O(n\log n)$ deterministic algorithm for 2-means clustering. The algorithm is fairly complicated and relies on several results in computational geometry that depend heavily on the dimensions, leading to a running time of $O(n\log n.\varepsilon^{-2d}\log(1/\varepsilon)+n\varepsilon^{-(4d-2)}\log(1/\varepsilon))$.

One of the questions left open in [9] by Matousek was whether $\Omega(n\log n)$ is a lower bound. We present a randomized algorithm with running time of $O((1/\varepsilon)^{O(1/\varepsilon)}(d/\varepsilon)^d n)$, which is linear in n.[2]

2. Preliminaries

We describe here some definitions and known results that we will use later in our algorithm.

2.1. Approximating the centroid of a set of points

The following lemma due to Inaba et al. [7] shows that the centroid of a set, P, of points in d-dimensional Euclidean space can be approximated by random sampling.

Lemma 2.1 [7]. *Let T be a sample of m points obtained by m independent draws at random from S. Then with probability $1-\delta$,*

$$\Delta\big(S, C(T)\big) < \left(1+\frac{1}{\delta m}\right)\Delta(S).$$

Thus, by considering the centroid of a random sample, S, of size $2/\varepsilon$ ($m=2/\varepsilon$), with constant probability ($\delta=1/2$), we can approximate the centroid of a set of points P, so that the cost of the clustering to this point is within a factor of $(1+\varepsilon)$ of the cost of the clustering to the mean of P. We call such an approximate, an *ε-approximation* to the centroid.

Note that this also establishes the existence of a set of $2/\varepsilon$ points of P, whose centroid is an ε-approximation to the centroid of P.

2.2. ε-near pairs

We present here the concept of ε-near pairs and a result on such pairs due to Matousek [9].

[2] In an independent work, Har-Peled and Mazumdar [6] discovered a linear time algorithm that runs in linear time for fixed k and in a more recent work [8] the authors describe a linear time algorithm for fixed k and *any* dimension.

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.4 (1-14)
by:violeta p. 4

Definition 2.1. For a real number $\varepsilon \geqslant 0$, we define a relation $\sim_\varepsilon$ on (ordered) pairs of points in d-dimensional Euclidean space, $\Re^d$: we let $(x, y) \sim_\varepsilon (x', y')$ if $\|x - x'\| \leqslant \varepsilon\|x - y\|$ and $\|y - y'\| \leqslant \varepsilon\|x - y\|$. We say that (x, y) and (x', y') are *ε-near* if $(x, y) \sim_\varepsilon (x', y')$ and $(x', y') \sim_\varepsilon (x, y)$. The relation $\sim_\varepsilon$ is not "quite" symmetric.

The following lemma is due to Matousek [9].

Lemma 2.2 [9]. *If (x, y) and (x', y') are ε-near pairs, and if C is the cost of the 2-means clustering induced by (x, y) and C' is the cost of the 2-means clustering induced by (x', y'), then,*

$$C \leqslant (1 + 16\varepsilon)C'.$$

This lemma states that if there are two pairs of centers that are close to each other, then the cost of the clusterings induced by the two pairs does not differ very much.

3. The algorithm

The main ideas behind the algorithm are as follows. We first obtain candidate centers for the centroid of the larger cluster by random sampling. We then construct a set of lines through each of these candidate centers to obtain a line close enough to the line joining the optimal centroid pair. We reduce the *approximate 2-means clustering* problem to a set of *center location on a line* problems determined by these lines. We then solve the approximate version of the *center location on a line* problem (defined below formally) on these lines by random sampling. This gives us an approximation to the centroid of the smaller cluster.

We define the center location on a line problem as follows:

Definition 3.1 (*Center location on a line*). Given a set P of n data points in d-dimensional Euclidean space, $\Re^d$, a fixed point c_1 and a line ℓ passing through c_1, the problem is to find another point c_2 on the line ℓ so as to minimize the cost of the 2-means clustering induced by (c_1, c_2). This measure, ϕ, is the cost of the center location on the line problem and is defined as

$$\phi(P, c_1, \ell) = \min_{c_2 \in \ell} \{\Delta(P_1, c_1) + \Delta(P_2, c_2)\}$$

where (P_1, P_2) is the partitioning induced by (c_1, c_2) (i.e., the partitioning induced by the perpendicular bisector of (c_1, c_2)).

The approximation version of this problem is stated as follows:

Definition 3.2 (*$(1 + \varepsilon)$-approximate center location on a line*). Given a set P of n data points in d-dimensional Euclidean space, $\Re^d$, a fixed point c_1, a line ℓ passing through c_1 and an approximation factor $\varepsilon > 0$, the problem is to determine another point c_2 on the line ℓ such that

$$\Delta(P_1, c_1) + \Delta(P_2, c_2) \leqslant (1 + \varepsilon)\phi(P, c_1, \ell)$$

where (P_1, P_2) is the partitioning induced by (c_1, c_2).

We also define a (p, θ)-covering as follows:

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.5 (1-14)
by:violeta p. 5

Definition 3.3 ((p, θ)-*covering*). Given a point p in d-dimensional Euclidean space, $\Re^d$, and a parameter θ ($0 < \theta < 2\pi$), we define a (p, θ)-covering to be a set of lines $\mathcal{Y}$ passing through p, such that, for any line ℓ, passing through p, there exists a line ℓ' in $\mathcal{Y}$ that makes an angle at most θ with ℓ.

The formal algorithm is described by the procedures 2-means and CenterLocation described in Figs. 1 and 2 respectively.

Algorithm 2-means reduces the *approximate* 2-*means clustering* problem to a set of *center location on a line* problems. For this, we start by randomly sampling a set of points, from which we can derive a set of candidate centers that contains an approximation to the centroid of the larger cluster, with constant probability. We proceed with the remaining steps for each of the candidate centers in the set. Now that we have an approximation to one of the centroids, our goal is to approximate the other centroid. We first approximate the line joining the optimal centers. For this, we construct a set of lines (covering) passing

---

Algorithm 2-**means**(P, ε)
Inputs : P: Point set, ε: approximation factor
Output : A $(1+\varepsilon)$ *approximate* 2-*means clustering* of the points in P

1. Let $\alpha = \varepsilon/16$, $\theta = \varepsilon/1024$, $\delta = \varepsilon/4$.
2. (Approximate the centroid of the larger cluster)
   (a) Randomly sample a set, R, of $8/\alpha$ points (independently drawn) from P.
   (b) Let T be the set of centroids of all subsets of R of size $2/\alpha$.
3. For each point c_1' in T,
   (a) Construct a set $\mathcal{Y}$ of lines that is a (c_1', θ)-covering.
   (b) For each ℓ in $\mathcal{Y}$, solve **CenterLocation**(P, c_1', ℓ, δ).
4. Return the solution which has minimum cost.

Fig. 1. The 2-means algorithm.

Algorithm CenterLocation(P, c_1, ℓ, δ)
Inputs : P: Point set, c_1: fixed center,
ℓ: line passing through c_1 on which to locate the second center
δ: approximation factor
Output : The solution for a $(1+\delta)$-*approximate center location on line* ℓ

1. Project the points of P onto line ℓ. Let P' be the projected points.
2. (Assume center to be located is to the right of c_1)
   For i = 1 to $\log |P'|$
   (a) Let M_R be the right most $|P'|/2^i$ points of P'.
   (b) Randomly sample a set, R of $8/\delta$ points (independently drawn) from M_R.
   (c) Let T be the set of centroids of all subsets of R of size $2/\delta$.
   (d) For each c_2' in T, compute the cost of the clustering induced by c_1, c_2'.
3. Repeat step 2 assuming the center to be located is to the left of c_1.
4. Return the solution which has minimum cost.

Fig. 2. The approximate center location on a line algorithm.

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.6 (1-14)
by:violeta p. 6

through the first center and making a small angle with each other, such that at least one of the lines is close enough, in orientation, to the actual line joining the optimal centers. For each of the lines in the covering above and the corresponding approximation to the centroid, we solve the *center location on a line problem*, by calling algorithm CenterLocation (we will later show that by restricting the other center to lie on this line, we only suffer by a small factor in our approximation).

Algorithm CenterLocation solves the *approximate center location on a line* problem on each of the candidate lines constructed above (we will later show that it suffices to solve the approximation version of this problem, suffering only by a small factor in our approximation). We first project all the points onto this line, converting it to a problem in one dimension after transformation. Since there are only two directions to the line, the smaller cluster must lie either to the right or to the left. We repeat the remaining steps for each of the two directions. We solve the problem, by guessing the size of the smaller cluster within a constant factor. For each of the $O(\log n)$ possible guesses, we again take a random sample from the points on that side of the line and obtain a set of points that contains an approximation to the centroid of the smaller cluster, with constant probability.

We obtain the costs of the clusterings for all possible pairs formed in this manner and report the clustering with the least cost. We will show that this entire procedure can be performed in time linear in n.

For ease of readability, we suppress the linear dependence on the dimension d when we analyze the algorithm, viz., linear time in our case means $O(nd)$ since each point has d coordinates. However, we do state the dependence on the terms depending exponentially on d.

The proofs of correctness for the algorithms are detailed in Section 4.

Remark 3.1. The centroid of a set of points, P in $\Re^d$, is the linear combination of the centroids of any partitioning of the point set. Having obtained an approximation to the centroid of the larger cluster of a 2-means clustering, say c, it would have been interesting if we could limit our search for the approximation to the other centroid to the line joining this centroid, c, to the centroid of the point set P. However, since we are dealing with the squares of the distances, it is not possible to get a decent bound on the approximation obtained by restricting ourselves to this line.

4. Proof of correctness

4.1. Reduction to approximate center location on lines

In this section we prove the correctness of the 2-means algorithm and show that we can reduce the *approximate* 2*-means clustering* problem to a set of *approximate center location on a line* problems so that the solution to one of these problems will correspond to a solution to the *approximate* 2*-means clustering* problem, with constant probability.

The following lemma shows that we can obtain an approximation to the centroid of the larger cluster by random sampling.

Lemma 4.1. *Let* (P_1, P_2) *be the optimal* 2*-means clustering of* P *and let* (c_1, c_2) *be the corresponding centroids. Without loss of generality, assume* $|P_1| \geqslant |P_2|$. *Let* R *be a random sample of size* $8/\alpha$ *obtained by independent draws at random from* P. *Let* T *be the set of centroids of all possible subsets of* R *of size*

$2/\alpha$. Then with constant probability, T contains a point, c_1', that is an α-approximation of the centroid, c_1, i.e., $\Delta(P_1, c_1') \leqslant (1+\alpha)\Delta(P_1, c_1)$.

Moreover, $|T| = O((1/\alpha)^{O(1/\alpha)})$.

Proof. As $|P_1| \geqslant |P_2|$, with constant probability, the sample R contains at least $2/\alpha$ points of P_1. Hence, by Lemma 2.1, with constant probability, T contains an α-approximation to the centroid, c_1, of P_1. Clearly, the size of T is $O((1/\alpha)^{O(1/\alpha)})$. □

The following lemma shows that if we have a point c_1' that is an α-approximation to the centroid c_1 of one of the clusters in an optimal 2-means clustering, then the cost of the optimal 2-means clustering induced by c_1' is within a factor of $(1+\alpha)$ of the cost of the optimal 2-means clustering.

Lemma 4.2. *Let (P_1, P_2) be the optimal 2-means clustering of P and let (c_1, c_2) be the corresponding centroids. Let c_1' be an α-approximation to the centroid c_1. Let c_2' be the other center in the optimal 2-means clustering (P_1', P_2') of P when center c_1' is fixed. Then*

$$\Delta(P_1', c_1') + \Delta(P_2', c_2') \leqslant (1+\alpha)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big).$$

Proof. As c_1' is an α-approximation to the centroid c_1, we have,

$$\Delta(P_1, c_1') + \Delta(P_2, c_2) \leqslant (1+\alpha)\Delta(P_1, c_1) + \Delta(P_2, c_2) \leqslant (1+\alpha)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big).$$

Also, by the optimality of the partitioning (P_1', P_2') when the center c_1' is fixed, we know that

$$\Delta(P_1', c_1') + \Delta(P_2', c_2') \leqslant \Delta(P_1, c_1') + \Delta(P_2, c_2).$$

Therefore,

$$\Delta(P_1', c_1') + \Delta(P_2', c_2') \leqslant (1+\alpha)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big). \qquad \square$$

The following lemma shows that when a center c_1 is fixed, and (P_1, P_2) is the optimal 2-means clustering induced by c_1, and c_2 is the centroid of P_2, then, instead of finding c_2, it suffices to find a suitable point on a line that makes a small angle with the line joining c_1 and c_2 at the cost of a small factor in our approximation.

Lemma 4.3. *Let c_1 and c_2 be points in d-dimensional Euclidean space, $\Re^d$. Let (P_1, P_2) be the clustering induced by (c_1, c_2). Let ℓ be the line joining c_1 and c_2. Let $0 < \beta \leqslant 1$ be a given constant. Let ℓ' be a line passing through c_1 that makes angle $\beta/64$ with ℓ. Let (P_1', P_3) be the optimal 2-means clustering induced by c_1 when the other center is also constrained to lie on ℓ'. Let c_3 be the optimal center corresponding to P_3. Further let c_3' be any other point lying on ℓ', such that,*

$$\Delta(P_1'', c_1) + \Delta(P_3', c_3') \leqslant \left(1 + \frac{\beta}{64}\right)\big(\Delta(P_1', c_1) + \Delta(P_3, c_3)\big) \tag{1}$$

where (P_1'', P_3') is the clustering induced by (c_1, c_3'). Then,

$$\Delta(P_1'', c_1) + \Delta(P_3', c_3') \leqslant (1+\beta)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big).$$

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.8(1-14)
by:violeta p. 8

Proof. Let c_2' be the projection of c_2 on ℓ' and let (P_1''', P_2') be the clustering induced by (c_1, c_2'). By the optimality of the centroids (c_1, c_3) on the line ℓ',

$$\Delta(P_1', c_1) + \Delta(P_3, c_3) \leqslant \Delta(P_1''', c_1) + \Delta(P_2', c_2'). \tag{2}$$

In the right triangle, $\triangle c_1c_2c_2'$, $\angle c_2c_1c_2' \leqslant \frac{\beta}{64}$ (see Fig. 1). Since $\tan\theta \leqslant 2\theta$ for $\theta \leqslant 1/2$, we have, $c_2c_2'/c_1c_2 \leqslant \beta/32$ and $c_2c_2'/c_1c_2' \leqslant \beta/32$. This implies that the pairs of points (c_1, c_2) and (c_1c_2') are $\beta/32$-near. Therefore, by Lemma 2.2,

$$\Delta(P_1''', c_1) + \Delta(P_2', c_2') \leqslant \left(1 + \frac{\beta}{2}\right)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big). \tag{3}$$

From Eqs. (1), (2) and (3), it follows that

$$\begin{aligned}\Delta(P_1'', c_1) + \Delta(P_3', c_3') &\leqslant \left(1 + \frac{\beta}{64}\right)\left(1 + \frac{\beta}{2}\right)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big)\\ &\leqslant (1+\beta)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big). \qquad \square\end{aligned}$$

The following lemma shows that given a point p, it is possible to construct a set of $O((d/\theta)^d)$ lines that is a (p, θ)-covering.

Lemma 4.4. *Given a point p in d-dimensional Euclidean space, $\Re^d$, and an approximation angle $0 < \theta \leqslant 1$, we can construct a set $\mathcal{Y}$ of $O((d/\theta)^d)$ lines passing through p, in $O((d/\theta)^d)$ time, such that given any line, ℓ, passing through p, there exists at least one line in $\mathcal{Y}$ that makes an angle at most θ with ℓ.*

Proof. Consider a unit d-hypercube centered at a point p. The d hypercube has $2d$ surfaces which are $(d-1)$-hypercubes. Consider the partitioning of each of these $(d-1)$-hypercubes into smaller equal $(d-1)$-hypercubes of side length x. Thus we get a set, $\mathcal{C}$ of $O((1/x)^{d-1})$ $(d-1)$-hypercubes.

Consider the set, $\mathcal{L}$, of lines passing through the corners of the hypercubes in $\mathcal{C}$ and the point p. Clearly the largest angle between any two lines is formed between the lines passing through the opposite two corners of the hypercube located at the center of any surface (side of the d-hypercube) as this is the closest $(d-1)$-hypercube to p. Thus in order to obtain a line covering of angle at most θ, it suffices to set x small enough to ensure that the angle between these pair of lines is less than θ.

It can be verified that setting $x = \theta/\sqrt{d}$ satisfies this condition, as $\tan\theta \geqslant \theta$ for $0 < \theta \leqslant 1$. Therefore we obtain a set, $\mathcal{L}$, of lines of size $O((d/\theta)^d)$.

Fig. 3 illustrates such a line covering for $d = 2$. The same can be extended to higher dimensions.

Clearly the time required to compute this set of lines is of the same order as its size. $\square$

The following lemma proves the correctness of the 2-means algorithm.

Lemma 4.5. *Given an algorithm* CenterLocation *that determines a $(1+\delta)$-approximate solution to the center location on a line problem in time $\mathcal{T}(n, \delta)$, with constant probability, where n is the number of input data points*; *the algorithm* 2-means *determines a $(1+\varepsilon)$-approximate 2-means clustering, with constant probability, in time $O((1/\varepsilon)^{O(1/\varepsilon)}(d/\varepsilon)^d.\mathcal{T}(n, \varepsilon/4))$.*

Proof. Let (P_1, P_2) be the optimal 2-means clustering and (c_1, c_2) be the corresponding centroids.

Step 1 of algorithm 2-means sets α, θ and δ to suitable constants (fractions of ε).

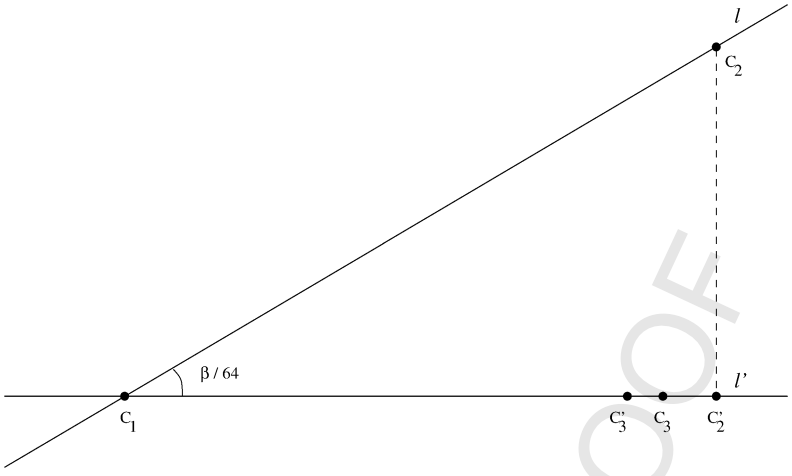

Fig. 3. Approximating the line connecting the centers. c_2' is the projection of c_2 on ℓ'. c_3 is the optimal on the line ℓ'.

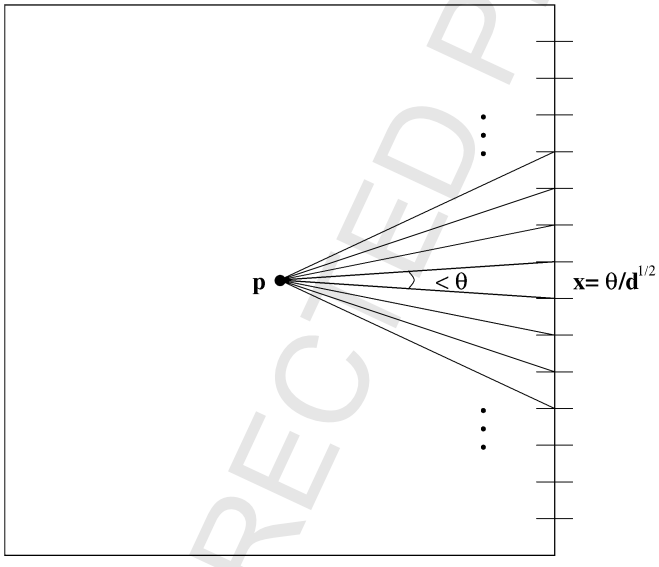

Fig. 4. A (p, θ)-covering in 2-dimensions ($d = 2$). (Not all the lines of the covering are shown in the figure.) The largest angle is formed with the hypercube at the center of the surface.

Applying Lemma 4.1, step 2 of the algorithm determines a set T of $O((1/\alpha)^{O(1/\alpha)})$ points, such that at least one of these points is an α-approximation to the centroid of the larger cluster, with constant probability. Let c_1' be such an α-approximation (if it exists).

Step 3 (a) of the algorithm determines a (p, θ)-covering $\mathcal{Y}$ for every center, p, in T (including c_1'). Applying Lemma 4.4, such a covering set of lines can be constructed in time $O((d/\theta)^d)$ and has size $O((d/\theta)^d)$. Let $\beta = 64\theta$. Then, we have $O((d/\beta)^d)$ lines passing through p, such that for any line passing through p, we have a line making an angle at most $\beta/64$.

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.10 (1-14)
by:violeta p. 10

In all, we have $O((d/\beta)^d)$ lines passing through each of the $O((1/\alpha)^{O(1/\alpha)})$ candidate centers, resulting in a total of $O((1/\alpha)^{O(1/\alpha)}(d/\beta)^d)$ lines. One of these lines, with constant probability, passes through c_1', and makes an angle of less than $\beta/64$ with the line joining c_1' and c_2 where c_2 is the other center of the optimal 2-means clustering, when the first center, c_1' is fixed.

Therefore, applying Lemmas 4.2 and 4.3, the solution to the $(1+\delta)$ *approximate center location on a line* problem on this line, computed in step $3-(b)$ of the algorithm, gives us a $(1+\alpha)(1+\beta)(1+\delta)$-*approximate* 2-*means clustering* of P. With the appropriate choices of α, β and δ (as specified in step 1 of the algorithm), we have a set of $O((1/\varepsilon)^{O(1/\varepsilon)}(d/\varepsilon)^d)$ lines, such that, the solution of the $(1+\varepsilon/4)$-*approximate center location on a line* problem on one of them gives us a solution to the $(1+\varepsilon)$-*approximate* 2-*means clustering* of P, with constant probability.

Since we are reporting the minimum cost clustering in step 4 of the algorithm, it is a $(1+\varepsilon)$- *approximate* 2-*means clustering*, with constant probability. □

4.2. Solving the problem for center location on a line

In this section we prove the correctness of the algorithm CenterLocation and show that we can solve the *approximate center location on a line* problem in linear time.

In the following lemma, we show that the center location on a line problem is reducible to one dimension.

Lemma 4.6. *Let P be a set of points in d-dimensional Euclidean space, $\Re^d$. Let ℓ be a given line and let c_1 be a fixed point on ℓ. Let P' be the set of points obtained by projecting the points of P on ℓ. Consider the problem $\mathcal{P}$, of locating a point c_2 on ℓ such that the clustering of P' induced by (c_1, c_2) is optimal when c_1 and c_2 are constrained to lie on ℓ with c_1 fixed.*

Then, the problem $\mathcal{P}$ is equivalent to the center location on a line problem. Moreover, an approximate solution to $\mathcal{P}$ is also an approximate solution to the center location on a line problem within the same approximation factor.

Proof. Consider the transformation of the original axis to an orthogonal axis in which one of the axis is parallel to ℓ. Let the point $p_i \in P$ be denoted by the vector $(x_{i1}, x_{i2}, \ldots, x_{id})$ on the transformed orthogonal axis with x_{i1} being the contribution towards the axis parallel to line ℓ. Let c_1 and c_2 be represented by the vectors $(c_{11}, c_{12}, \ldots, c_{1d})$ and $(c_{21}, c_{22}, \ldots, c_{2d})$ respectively on the transformed axis. Then the cost of partition (P_1, P_2) with centers (c_1, c_2) can be computed as

$$\Delta = \sum_{p_i \in P_1} \sum_{j=1}^{d} (x_{ij} - c_{1j})^2 + \sum_{p_i \in P_2} \sum_{j=1}^{d} (x_{ij} - c_{2j})^2.$$

Now, since c_1 lies on ℓ and c_2 is also constrained to lie on ℓ, therefore minimization of the above cost function can be reduced to the axis parallel to ℓ as follows:

$$\Delta_1 = \sum_{p_i \in P_1} (x_{i1} - c_{11})^2 + \sum_{p_i \in P_2} (x_{i2} - c_{22})^2.$$

This is because, the difference in the cost of a point $p_i \in P$ towards two distinct centers constrained to lie on line ℓ is only affected by the component along the axis parallel to ℓ. The cost incurred along all the remaining axis remains unaffected.

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.11(1-14)
by:violeta p. 11

Moreover, we also claim that a $(1+\delta)$-approximate clustering on the single dimension problem corresponds to a $(1+\delta)$-approximate clustering in the original space.

Let (P_1', P_2') be a $(1+\delta)$-approximate clustering in a single dimension (axis parallel to ℓ) with centers (c_1, c_2') both lying on ℓ. Then

$$\Delta\big((P_1', c_1) + \Delta(P_2', c_2')\big) \leqslant (1+\delta)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big),$$

i.e.,

$$\sum_{p_i \in P_1'} (x_{i1} - c_{11})^2 + \sum_{p_i \in P_2'} (x_{i2} - c_{22}')^2 \leqslant (1+\delta)\Bigg(\sum_{p_i \in P_1} (x_{i1} - c_{11})^2 + \sum_{p_i \in P_2} (x_{i2} - c_{22})^2\Bigg).$$

Therefore,

$$\sum_{p_i \in P_1'} \sum_{j=1}^{d} (x_{ij} - c_{1j})^2 + \sum_{p_i \in P_2'} \sum_{j=1}^{d} (x_{ij} - c_{2j}')^2$$
$$\leqslant (1+\delta)\Bigg(\sum_{p_i \in P_1} \sum_{j=1}^{d} (x_{ij} - c_{1j})^2 + \sum_{p_i \in P_2} \sum_{j=1}^{d} (x_{ij} - c_{2j})^2\Bigg)$$

as components along all other dimensions are unaffected. □

We state here some observations that will be useful in our next proof.

Observation 4.1. Given two points (c_1, c_2), let $\mathcal{H}$ be the perpendicular bisector of these points. This hyperplane gives the optimal clustering with fixed centers (c_1, c_2). As we move a hyperplane parallel to $\mathcal{H}$ in either direction, the cost function of the clustering due to the hyperplane with fixed centers (c_1, c_2) is a monotonic non-decreasing function. This is clear since, as we move the hyperplane, the cost remains unchanged till we cross a point. When we cross a point, we disassociate this point from its closer center and associate it to the further center and this can only result in an increase in the cost of the clustering.

Observation 4.2. Given a set of points, along with the centroid of the set and the cost of the cluster, on addition or removal of a point to/from this set, we can compute the new centroid and the cost of the new cluster in constant time. If $\mathbf{x}$ is the old centroid of a set P of n points and $\mathbf{x}_{n+1}$ is a new point, so that $P' = P \cup \{\mathbf{x}_{n+1}\}$ then the new centroid $\mathbf{x}_{\text{new}}$ is obtained as follows:

$$\mathbf{x}_{\text{new}} = \frac{n}{n+1}\mathbf{x} + \frac{1}{n+1}\mathbf{x}_{n+1},$$

and the cost of the clustering at $\mathbf{x}_{\text{new}}$ is obtained as follows:

$$\Delta(P', \mathbf{x}_{\text{new}}) = \Delta(P, \mathbf{x}) + n\Delta(\mathbf{x}, \mathbf{x}_{\text{new}}) + \Delta(\mathbf{x}_{n+1}, \mathbf{x}_{\text{new}}).$$

It follows that given a 2-partitioning of P along with the centroids and the cost of the clustering due to the two partitions, if we move one point from one partition to another partition, then the cost of the new clustering and the new centroids can be determined in constant time.

Remark 4.1. The Center Location on a line problem is solvable exactly by traversing the points from left to right one at a time and updating the cost of the clustering using the above observation. This can

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.12(1-14)
by:violeta p. 12

be done in two passes, first assuming the center lies to the right and then assuming that it lies to the left. However, note that this requires $O(n \log n)$ time as the points have to be sorted before the traversal can be performed. Our goal is to achieve linear running time.

In the following lemma, we prove the correctness of the algorithm CenterLocation and determine its running time.

Lemma 4.7. *Let P be a set of points in d-dimensional Euclidean space, $\Re^d$, and ℓ be a given line. Let c_1 be a fixed point on ℓ. Then, algorithm* CenterLocation *determines in time $O((1/\delta)^{O(1/\delta)}n)$, a point c_2' such that with constant probability, the cost of the clustering induced by (c_1, c_2') is within a factor of $(1+\delta)$ of the cost of the clustering induced by (c_1, c_2), i.e.,*

$$\Delta(P_1', c_1) + \Delta(P_2', c_2') \leqslant (1+\delta)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big)$$

where c_2 is the point on ℓ for which the clustering of P induced by (c_1, c_2) is optimal when c_1 is fixed, (P_1, P_2) is the corresponding partitioning of the points and (P_1', P_2') is the partitioning of P induced by (c_1, c_2').

Proof. Step 1 of algorithm CenterLocation projects the points P onto the line ℓ. Applying Lemma 4.6, the problem of solving the *approximate center location on a line* problem is equivalent to solving the approximate version of the problem after projecting the points on ℓ.

Step 2 of the algorithm assumes the center of the smaller cluster lies to the right of c_1 (The case when the center lies to the left of c_1 is handled similarly in step 3.) In each iteration half the remaining points from the left side are eliminated as in step 2(a) of the algorithm. Suppose in the current iteration we have a set M of m points. Initially $m = n$. We first locate the mid-point in M, i.e., $m/2$th point, say q. This takes $O(m)$ time and partitions the point set, M, into two sets M_L and M_R. M_R is the remaining points after elimination. Thus, M_R can be obtained from the points remaining in the previous iteration in $O(m)$ time.

Clearly, in some iteration, the number of points remaining, $|M_R|$, will be at most twice the size of P_2 and P_2 will be contained in M_R. Also, since the number of points are being halved in each iteration, the number of iterations required is at most $\log n$.

Assuming that we have the right partition, in step 2(b) of the algorithm, we randomly sample a set R of $8/\delta$ points from M_R. Note that although we only require $2/\delta$ points to approximate the centroid (Lemma 2.1), we take ample points in order to sample $2/\delta$ points from P_2 with constant probability, since we are approximating its size within a factor of 2. In step 2(c) of the algorithm, we now take all possible subsets of the random sample R of size $2/\delta$ and compute their centroids. Note that there are $O((1/\delta)^{O(1/\delta)})$ such points. With constant probability, one of these is a δ-approximation to the centroid of P_2, i.e.,

$$\Delta(P_2, c_2') \leqslant (1+\delta)\Delta(P_2, c_2).$$

Therefore,

$$\begin{aligned}\Delta(P_1', c_1) + \Delta(P_2', c_2') &\leqslant \Delta(P_1, c_1) + \Delta(P_2, c_2') \leqslant \Delta(P_1, c_1) + (1+\delta)\Delta(P_2, c_2)\\ &\leqslant (1+\delta)\big(\Delta(P_1, c_1) + \Delta(P_2, c_2)\big).\end{aligned}$$

In step 2(d) of the algorithm, we compute the cost of the clustering. We now show that the clustering cost can be computed in time linear in the number of remaining points. In each iteration, we are dealing

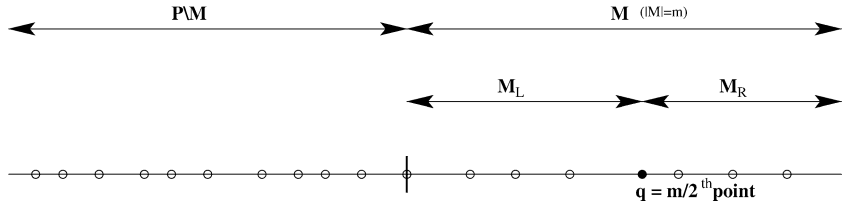

Fig. 5. Sampling recursively by guessing the size of $|P_2|$.

with half the points from the previous iteration. Let q be the mid point of M as above. The cost of the clustering obtained by partitioning at q can be determined in $O(m)$ time. This is possible because we already know the cost of the clustering due to $(M, P \setminus M)$ from the previous iteration and therefore we can partition the set M into M_L and M_R in $O(m)$ time and then apply Observation 4.2 over the set M_L. Note that in the current iteration we perform the sampling necessary for the guess $m/4 \leqslant |P_2| \leqslant m/2$. Let $\mathcal{H}'$ be the corresponding perpendicular bisecting line. If $\mathcal{H}'$ lies to the right of q, as we have already computed the cost of the clustering due to the partitioning at q, we only need to alter the cost over the remaining points in M_R which is at most $m/2$ and can be done in time $O(m)$ by applying Observation 4.2 at most $m/2$ times. If $\mathcal{H}'$ is to the left of q, let $\mathcal{H}$ be the optimal line bisecting the line joining the centroids (c_1, c_2). In our current guess, $m/4 \leqslant |P_2| \leqslant m/2$. Therefore, the line $\mathcal{H}$ must lie to the right of q. Due to Observation 4.1, the cost of the clustering due to the line $\mathcal{H}''$, passing through q and parallel to $\mathcal{H}$, must be between that of $\mathcal{H}$ and $\mathcal{H}'$. Thus, if $\mathcal{H}'$ defines a $(1+\delta)$-*approximate* 2-*means clustering*, then so does $\mathcal{H}''$ for which we already know the cost. Hence we have restricted ourselves to altering the cost known at q only over the set M_R for each of the candidate centers. Hence the time required to compute the cost of the clustering in each iteration is $O(m)$.

As we are always recurring into the right half, the running time of this algorithm is characterized by the recurrence relation, $T(n) = T(n/2) + cn$, which is linear in n. Performing the same steps in the other direction in step 3 of the algorithm at most doubles the running time.

In step 4, we return the minimum of the costs of the clusterings due to the pairs formed from these centers with c_1 fixed. This gives us a $(1+\delta)$-approximation to the *center location on the line* problem, with constant probability. □

Putting everything together, we obtain the main result of the paper:

Theorem 4.8. *Algorithm* 2-means, *using algorithm* CenterLocation, *determines a* $(1+\varepsilon)$-*approximate* 2-*means clustering of a point set* P *in* $\Re^d$ *in time* $O((1/\varepsilon)^{O(1/\varepsilon)}(d/\varepsilon)^d n)$, *with constant probability.*

Proof. By Lemma 4.7, algorithm CenterLocation solves the $(1+\delta)$-*approximate center location on a line* problem in time $O((1/\delta)^{O(1/\delta)}n)$.

Therefore, applying Lemma 4.5 and using the fact that the algorithm CenterLocation is invoked with $\delta = \varepsilon/4$, we conclude that the algorithm 2-means solves the *approximate* 2-*means clustering* problem in time $O((1/\varepsilon)^{O(1/\varepsilon)}(d/\varepsilon)^d n)$. □

S0925-7721(05)00014-3/FLA AID:780 Vol.●●●(●●●)
COMGEO:m2 v 1.36 Prn:20/04/2005; 8:11
cgt780
[DTD5] P.14 (1-14)
by:violeta p. 14

Acknowledgement

The authors would like to thank Pankaj Agarwal for his useful feedback regarding improving the readability of the paper.

References

[1] A. Broder, S. Glassman, M. Manasse, G. Zweig, Syntactic clustering of the web, in: Proc. of 6th International World Wide Web Conference, 1997.
[2] S.C. Deerwester, S.T. Dumais, T.K. Landauer, G.W. Furnas, R.A. Harshman, Indexing by latent semantic analysis, J. Amer. Soc. Inform. Sci. 41 (6) (1990) 391–407.
[3] Drineas, Frieze, Kannan, Vempala, Vinay, Clustering in large graphs and matrices, in: SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms), 1999.
[4] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, W. Equitz, Efficient and effective querying by image content, J. Intell. Inf. Syst. 3 (3–4) (1994) 231–262.
[5] M.R. Garey, D.S. Johnson, Computer and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, New York, 1979.
[6] S. Har-Peled, S. Mazumdar, On coresets for k-means and k-median clustering, in: Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, ACM Press, New York, 2004, pp. 291–300.
[7] M. Inaba, N. Katoh, 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, ACM Press, New York, 1994, pp. 332–339.
[8] A. Kumar, Y. Sabharwal, S. Sen, A simple linear time $(1+\varepsilon)$-approximation algorithm for k-means clustering in any dimensions, in: Proceedings of the 45th Annual Symposium on Foundations of Computer Science, IEEE Computer Society, 2004, pp. 454–462.
[9] J. Matousek, On approximate geometric k-clustering, Discrete Comput. Geom. 24 (1) (2000) 61–84.
[10] N. Megiddo, K.J. Supowit, On the complexity of some common geometric location problems, SIAM J. Comput. 13 (1) (1984) 182–196.
[11] M.J. Swain, D.H. Ballard, Color indexing, Internat. J. Computer Vision (1991) 11–32.