Pattern Recognition 43 (2010) 2698–2711

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Quantization-based clustering algorithm

Zhiwen Yu [a,b], Hau-San Wong [b,*]

[a] School of Computer Science and Engineering, South China University of Technology, China
[b] Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 4 May 2009
Received in revised form
6 February 2010
Accepted 24 February 2010

Keywords:
Histogram
Clustering algorithm
K-means

ABSTRACT

In this paper, a quantization-based clustering algorithm (QBCA) is proposed to cluster a large number of data points efficiently. Unlike previous clustering algorithms, QBCA places more emphasis on the computation time of the algorithm. Specifically, QBCA first assigns the data points to a set of histogram bins by a quantization function. Then, it determines the initial centers of the clusters according to this point distribution. Finally, QBCA performs clustering at the histogram bin level, rather than the data point level. We also propose two approaches to improve the performance of QBCA further: (i) a shrinking process is performed on the histogram bins to reduce the number of distance computations and (ii) a hierarchical structure is constructed to perform efficient indexing on the histogram bins. Finally, we analyze the performance of QBCA theoretically and experimentally and show that the approach: (1) can be easily implemented, (2) identifies the clusters effectively and (3) outperforms most of the current state-of-the-art clustering approaches in terms of efficiency.

1. Introduction

As is well known, clustering is one of the classical problems in many different applications [1–9], such as pattern recognition, multimedia, data mining, knowledge discovery and data compression. The objective of clustering is to partition the data into different groups. Data which are in the same group have higher similarity, while data which originate from different groups have lower similarity. Based on various criteria, a large number of approaches for finding good clusters have been proposed in different applications [48], such as clustering based on hierarchical relationship [1–5], clustering based on density [6,7], clustering based on a grid structure [8,9], and clustering based on partition [10–30]. One of the most important techniques is K-means and its variants [10–30] in which a well-defined objective function is minimized. Suppose (i) the data set P consists of n data points ($P=\{\mathbf{p}_1,\ldots,\mathbf{p}_n\}$) and (ii) each data point $\mathbf{p}_i$ has m attributes ($\mathbf{p}_i=\{p_{i\,1},\ldots,p_{im}\}$), the objective function of K-means is to search for a set of cluster centers $S=\{\mathbf{s}_1^*,\mathbf{s}_2^*,\ldots,\mathbf{s}_k^*\}$ which minimize the sum of squared error:

$$\{\mathbf{s}_1^*,\mathbf{s}_2^*,\ldots,\mathbf{s}_k^*\}=\arg\min_{\{\mathbf{s}_1,\mathbf{s}_2,\ldots,\mathbf{s}_k\}\subset\mathbb{R}^m}\sum_{h=1}^{k}\sum_{\mathbf{p}_i\in P_h}\|\mathbf{s}_h-\mathbf{p}_i\|^2 \quad (1)$$

Compared with other clustering algorithms, the K-means family of algorithms has three major advantages: (i) simple implementation; (ii) efficient when handling a large data set; and (iii) a solid theoretical foundation based on the greedy optimization of the Voronoi partition. As a result, K-means is widely used in real world applications.

Some of these applications require the processing of a very large number of data points. For example, during the process of image segmentation, hundreds of thousands of pixels in an image are required to be separated into several groups [33-36]. Each pixel is viewed as a point consisting of multiple attributes, such as the RGB color values. As another example, with the pervasiveness of mobile devices and wide availability of wireless networks, a large number of location data are generated by the mobile users. Although traditional K-means and its variants can be applied to cluster these data, most of them are not very effective in handling this kind of large data set.

In this paper, we propose a quantization-based clustering algorithm (QBCA) for classifying a large number of data points, which processes large sets of data points in an efficient and effective way. Compared with traditional clustering algorithms, QBCA achieves low computation time without sacrificing the quality of the clusters. The approach is composed of three steps. (1) After assigning all the data points to a set of histogram bins by a quantization function, QBCA first selects the initial centers of the clusters according to the number of points assigned to each histogram bin. (2) Then, it makes use of specific properties of the minimum and maximum values of a distance measure to perform clustering at the histogram bin level. (3) Finally, QBCA assigns the points to their corresponding clusters. The major idea of QBCA is to perform clustering at the histogram bin level, rather than the data point level. Two approaches are proposed to improve the performance of QBCA further: (i) a shrinking process is applied to

* Corresponding author. Tel.: +852 27888624.
E-mail address: cshswong@cityu.edu.hk (H.-S. Wong).

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.02.020

the histogram bins to reduce the number of distance computations and (ii) a hierarchical structure is constructed to perform efficient indexing on the histogram bins. We compare QBCA with traditional clustering algorithms, which are variants of the K-means algorithm, on synthetic data sets and real data sets. The results of the experiments show that QBCA outperforms most of the current algorithms in terms of efficiency.

The contribution of the paper is twofold. First, we design a quantization-based clustering algorithm (QBCA) to classify a large number of data points. Second, we propose two approaches, namely the shrinking process and the hierarchical structure, to improve the performance of QBCA further.

The remainder of the paper is organized as follows. Section 2 describes previous related works on clustering algorithms. Section 3 provides the definitions of a number of new terms related to the proposed approach. Section 4 introduces the quantization-based clustering algorithm (QBCA). Section 5 describes the shrinking process for improving the performance of QBCA. Section 6 describes the hierarchical structure for histogram bin indexing. Section 7 evaluates our proposed algorithms experimentally, and Section 8 provides our conclusion and describes future research directions.

2. Related work

Although there are many sophisticated clustering algorithms which are recently proposed [1–9], such as CURE [1], CHAMELEON [2], ROCK [3], GRIN [4], BIRCH [5], DBSCAN [6], OPTICS [7], STING [8], CLIQUE [9], the most popular algorithms for performing clustering on the data set with a large number of data points remain those belonging to the K-means family [10–30]. Due to their efficiency and effectiveness to handle large sets of data points, the algorithms in the K-means family are very important for efficient clustering. Since our proposed algorithm belongs to the K-means family, we only consider these algorithms in our subsequent description.

K-means partitions the data points into k clusters to minimize an objective function $f(\mathbf{S},\mathbf{W})$.

$$f(\mathbf{S},\mathbf{W}) = \sum_{h=1}^{k}\sum_{i=1}^{n}\sum_{l=1}^{m} \omega_{i,h} \cdot d(s_{h,l},p_{i,l}) \tag{2}$$

subject to

$$\sum_{h=1}^{k} \omega_{i,h} = 1 \tag{3}$$

where $\mathbf{S}$ is a set of cluster centers ($\mathbf{S}=\{\mathbf{s}_1, \ldots, \mathbf{s}_k\}$), and $d(s_{h,l},p_{i,l})$ denotes the distance between the center $\mathbf{s}_h$ of the h th cluster and the data point $\mathbf{p}_i$ in the l-th attribute. $\mathbf{W}$ is an $n \times k$ partition matrix, and $\omega_{i,h}$ is its constituent variable: If $\omega_{i,h}=1$, the data point $\mathbf{p}_i$ belongs to the h th cluster.

If the attribute is numeric,

$$d(s_{h,l},p_{i,l}) = (s_{h,l}-p_{i,l})^2 \tag{4}$$

If the attribute is categorical,

$$d(s_{h,l},p_{i,l}) = \begin{cases} 1 & \text{if } p_{i,l} \neq s_{h,l} \\ 0 & \text{if } p_{i,l} = s_{h,l} \end{cases} \tag{5}$$

Clearly, if all attributes in the data are categorical, K-means is equivalent to its variant K-modes [20]. On the other hand, if the data contains both categorical values and numerical values, K-means becomes equivalent to its variant K-prototypes [21].

In particular, K-means can be considered as a special case of the Expectation–Maximization (EM) algorithm [31]. As a result, the process of K-means mainly consists of two steps. In the expectation step, K-means fixes the matrix $\mathbf{S}$, which is the set of cluster centers, and determines the matrix $\mathbf{W}$ as follows:

$$d(\mathbf{s}_{h^*},\mathbf{p}_i) = \min_{h=1\ldots k} d(\mathbf{s}_h,\mathbf{p}_i) \tag{6}$$

$$\omega_{i,h} = \begin{cases} 1 & \text{if } h = h^* \ \ \forall h \in \{1,\ldots, \ \ k\} \\ 0 & \text{if } h \neq h^* \end{cases} \tag{7}$$

In the maximization step, K-means fixes the matrix $\mathbf{W}$, and determines the matrix $\mathbf{S}$ as follows:

$$\mathbf{s}_h = \frac{\sum_{i=0}^{n_h} \omega_{i,h}\mathbf{p}_i}{\sum_{i=0}^{n_h} \omega_{i,h}} \quad \forall h \in \{1,\ldots,k\} \tag{8}$$

where n_h denotes the number of data points which are assigned to the h th cluster.

Recently, there are a lot of research works which focus on how to improve the performance of K-means. For example, Laszlo et al. [41] proposed to adopt a genetic algorithm (GA) to search for the centers of K-means with the help of a hyper-quadtree constructed on the data, in order to obtain the global optimal value of the objective function in K-means. They also designed a novel crossover operator that exchanges neighboring centers during the genetic optimization process [42], and selected cluster centers as the initial seeds for K-means. Bandyopadhyay et al. [43] designed a new point symmetry-based distance measure to improve the performance of K-means. Chung et al. [44] proposed the modified point symmetry-based K-means (MPSK) algorithm which works well for both the symmetrical intra-clusters and the symmetrical inter-clusters. Lai et al. [45] partitioned cluster centers into inactive and active sets based on the information of cluster displacement. They reduced the computational complexity of K-means by only focusing on cluster centers in the active set. Lai et al. [46] further improved the efficiency of K-means by the displacement of cluster centers to reject useless candidates for a data point. Chang et al. [47] improved the genetic algorithm based K-means by adopting adaptive probabilities for the crossover and mutation operators.

Although there are different variants of K-means [20–30], one of the most efficient algorithms in the K-means family is the filtering algorithm which is based on the kd-tree [10]. The filtering algorithm (i) stores the data points in a kd-tree and (ii) maintains a subset of candidate centers for each node of the kd-tree. When the candidates are propagated to the node's children, some of them are pruned. The lower the level the node is at, the smaller the number of candidates the node contains. During the K-means clustering process, we do not need to consider all the k centers, but only those candidate centers in the node which are associated with the current data point. In other words, the filtering algorithm achieves low computation time by reducing the distance computation cost. Based on the filtering algorithm, different K-means variants are proposed, which include: (i) the swap algorithm (SWAP), which applies perturbation to the current set of cluster centers to escape from local minima. Specifically, selected centers are swapped between the current center set and a reserved list of candidate centers; (ii) the simple hybrid algorithm (EZ-Hybrid), which performs a single swap followed by a pre-specified number of K-means iterations; and (iii) the Hybrid algorithm (Hybrid), which performs a selected number of swaps followed by multiple K-means iterations. However, the algorithm needs $O(n \log n)$ time complexity to construct a kd-tree, and the resulting computation time cannot satisfy the requirement of efficient clustering, especially when the set of pixels is large. The motivation of this paper is to reduce the clustering time further by substituting the kd-tree with a histogram-based indexing structure, since the time complexity of constructing a histogram by a suitable quantization function is $O(n)$.

In addition, K-means has two limitations: (i) The initial centers influence the performance of the algorithm and (ii) the computation time of the algorithm is proportional to the distance computations between the data points and the centers of the clusters. In view of these limitations, we propose the current quantization-based clustering algorithm.

3. Term definition

In this section, a number of terms related to our proposed quantization-based clustering algorithm (QBCA) are formalized.

First, the seed set which contains the cluster centers is defined as follows:

Definition 1 (*Seed set*). The seed set S consists of k seeds which serve as the initial cluster centers.

$$S=\{\mathbf{s}_h\}_{h=1}^{k} \quad (\mathbf{s}_h \in \mathbb{R}^m) \tag{9}$$

Then, QBCA performs clustering at the histogram bin level (where the histogram bin is defined as a uniformly quantized input space). To facilitate the development of QBCA, two distance metrics (the minimum distance $\underline{d}$ and the maximum distance $\overline{d}$) are defined as follows:

Definition 2 (*Minimum distance*). The minimum distance $\underline{d}(\mathbf{s}_h,\mathbf{b}_j)$ between a seed $\mathbf{s}_h$ and a histogram bin $\mathbf{b}_j$ in the space $\mathbb{R}^m$ is

$$\underline{d}(\mathbf{s}_h,\mathbf{b}_j)=\sqrt{\sum_{l=1}^{m}(s_{h,l}-b_{j,l})^2} \tag{10}$$

$$b_{j,l}=\begin{cases}\underline{b}_{j,l} & \text{if } s_{j,l}<\underline{b}_{j,l}\\ \overline{b}_{j,l} & \text{if } s_{j,l}>\overline{b}_{j,l}\\ s_{h,l} & \text{otherwise}\end{cases} \tag{11}$$

where $b_{j,l}$ is the value of the histogram bin $\mathbf{b}_j$ in the l-th dimension, and $\underline{b}_{j,l}$ and $\overline{b}_{j,l}$ are the minimum and maximum coordinates of the histogram bin $\mathbf{b}_j$ in the l-th dimension, respectively.

Definition 3 (*Maximum distance*). The maximum distance $\overline{d}(\mathbf{s}_h,\mathbf{b}_j)$ between a seed $\mathbf{s}_h$ and a histogram bin $\mathbf{b}_j$ in the space $\mathbb{R}^m$ is

$$\overline{d}(\mathbf{s}_h,\mathbf{b}_j)=\sqrt{\sum_{l=1}^{m}(s_{h,l}-\Upsilon_{j,l})^2} \tag{12}$$

$$\Upsilon_{j,l}=\begin{cases}\underline{b}_{j,l} & \text{if } s_{h,l}\geq\dfrac{\underline{b}_{j,l}+\overline{b}_{j,l}}{2}\\ \overline{b}_{j,l} & \text{otherwise}\end{cases} \tag{13}$$

Fig. 1 illustrates the minimum distance $\underline{d}(\mathbf{s}_h,\mathbf{b}_j)$ and the maximum distance $\overline{d}(\mathbf{s}_h,\mathbf{b}_j)$ between the seed $\mathbf{s}_h$ and the bin $\mathbf{b}_j$ in a 2D histogram. For example, since $s_{h,1}>\overline{b}_{j,1}$ and $s_{h,2}>\overline{b}_{j,2}$, the minimum distance $\underline{d}(\mathbf{s}_h,\mathbf{b}_j)$ between the seed $\mathbf{s}_h$ and the bin $\mathbf{b}_j$ is equal to the distance between the seed $\mathbf{s}_h$ and the point in the upper right corner of the bin $\mathbf{b}_j$. Since $s_{h,1}>\underline{b}_{j,1}+\overline{b}_{j,1}/2$ and $s_{h,2}>\underline{b}_{j,2}+\overline{b}_{j,2}/2$, the maximum distance $\overline{d}(\mathbf{s}_h,\mathbf{b}_j)$ between the seed $\mathbf{s}_h$ and the bin $\mathbf{b}_j$ is equal to the distance between the seed $\mathbf{s}_h$ and the point in the lower left corner of the bin $\mathbf{b}_j$.

Definition 4 (*Seed candidate of the histogram bin*). If a seed $\mathbf{s}$ satisfies the following condition:

$$\underline{d}(\mathbf{s},\mathbf{b})\leq\overline{d}^{*}(\mathbf{s},\mathbf{b}) \tag{14}$$

$$\overline{d}^{*}(\mathbf{s},\mathbf{b})=\min_{1\leq h\leq k}\ \overline{d}(\mathbf{s}_h,\mathbf{b}) \tag{15}$$

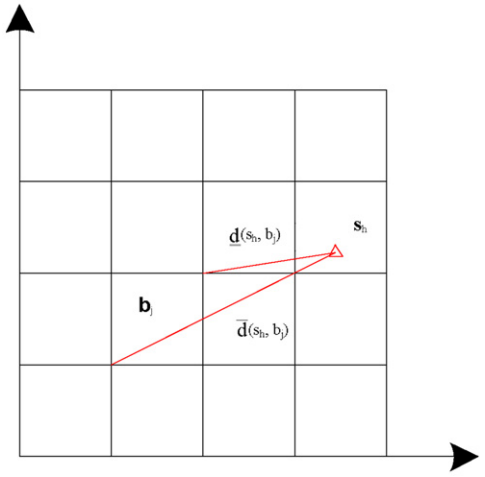


Fig. 1. An example of minimum distance and maximum distance.

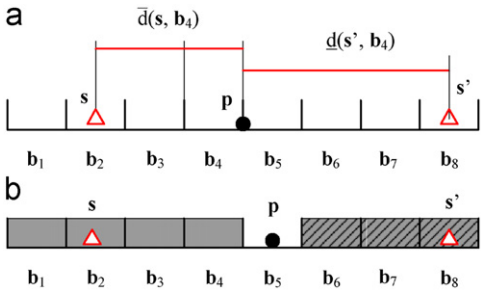


Fig. 2. An example to illustrate Lemma 1.

$\mathbf{s}$ is a seed candidate of the histogram bin $\mathbf{b}$. The seed candidate list $\mathbf{b}.list$ of the histogram bin $\mathbf{b}$ contains all the seed candidates which satisfy the above conditions.

$$\mathbf{b}.list=\{\mathbf{s}\in\mathbf{S}|\underline{d}(\mathbf{s},\mathbf{b})\leq\overline{d}^{*}(\mathbf{s},\mathbf{b})\} \tag{16}$$

The seed candidates of the histogram bin $\mathbf{b}$ are obtained based on the following two metrics: the minimum distance $\underline{d}$ between the seed and the histogram bin, and the maximum distance $\overline{d}$ between the seed and the histogram bin. Based on properties of these two measures, we have the following lemma.

Lemma 1. *If* $\underline{d}(\mathbf{s}',\mathbf{b})>\overline{d}^{*}(\mathbf{s},\mathbf{b})$, *then* $\mathbf{s}'$ *is not the closest seed of a point* $\mathbf{p}$ *which belongs to the histogram bin* $\mathbf{b}$.

The proof of Lemma 1 is given in Appendix A.

Obviously, there exists at least one seed $\mathbf{s}$ whose maximum distance to the histogram bin $\mathbf{b}$ is equal to $\overline{d}^{*}(\mathbf{s},\mathbf{b})$ ($\overline{d}^{*}(\mathbf{s},\mathbf{b})=\overline{d}(\mathbf{s},\mathbf{b})$). Then, the seed $\mathbf{s}$ is closer to the point $\mathbf{p}$ than $\mathbf{s}'$. This implies that it will not be possible for the seed $\mathbf{s}'$ to be the closest seed of the point $\mathbf{p}$, if $\mathbf{p}$ belongs to the histogram bin $\mathbf{b}$. As a result, the seed $\mathbf{s}'$ is not the seed candidate of the histogram bin $\mathbf{b}$. Fig. 2(a) gives an example of Lemma 1. The position of the point $\mathbf{p}$ is on the right boundary of the histogram bin $\mathbf{b}_4$. The distance $d(\mathbf{s}',\mathbf{p})$ between the seed $\mathbf{s}'$ and the point $\mathbf{p}$ is equal to $\underline{d}(\mathbf{s}',\mathbf{b}_4)$, while the distance $d(\mathbf{s},\mathbf{p})$ between $\mathbf{s}$ and $\mathbf{p}$ is $\overline{d}^{*}(\mathbf{s},\mathbf{b})$, which is equal to $\overline{d}(\mathbf{s},\mathbf{b}_4)$. Then, $d(\mathbf{s},\mathbf{p})=\overline{d}^{*}(\mathbf{s},\mathbf{b})<\underline{d}(\mathbf{s}',\mathbf{b}_4)=d(\mathbf{s}',\mathbf{p})$. As a result, the seed $\mathbf{s}'$ cannot be the closest seed to $\mathbf{p}$, if $\mathbf{p}$ belongs to the histogram bin $\mathbf{b}_4$. The seed candidate lists of the histogram bins are shown in Fig. 2(b). The gray histogram bins without the slashes contain the seed $\mathbf{s}$ as the candidate in their seed candidate lists, while the gray histogram bins with the slashes include the seed $\mathbf{s}'$ as the candidate in their seed candidate lists. The white

histogram bin $\mathbf{b}_5$ adds both the seeds $\mathbf{s}$ and $\mathbf{s}'$ to their seed candidate lists, since it is possible for $\mathbf{s}$ and $\mathbf{s}'$ to become the closest seed to the point $\mathbf{p}$ according to the relationship between the position of $\mathbf{p}$ and the positions of the boundaries of the histogram bin $\mathbf{b}_5$.

Based on the property of the seed candidate list of a histogram bin, the following lemma is obtained.

Lemma 2. *For an arbitrary point* $\mathbf{p}$ *which belongs to the histogram bin* $\mathbf{b}$, *the closest seed* $\mathbf{s}$ *to the point* $\mathbf{p}$ *is in the seed candidate list* $\mathbf{b}$.*list of the histogram bin* $\mathbf{b}$.

The proof of Lemma 2 is given in Appendix B.

According to Lemma 2, it is not necessary to compute the distance between the point $\mathbf{p}$ and all the seeds in order to obtain the closest seed to $\mathbf{p}$. Only the seeds in the seed candidate list of the histogram bin $\mathbf{b}$ which contains the point $\mathbf{p}$ need to be considered.

4. Quantization-based clustering algorithm

The objective of QBCA is to reduce the distance computation cost in the clustering process. Fig. 3 illustrates the framework of QBCA. As indicated in the figure, we first perform quantization to associate the data points with their corresponding histogram bins. Then, we apply a step known as the cluster center initialization stage (CCI) to find the "good" initial centers of the clusters. Finally, the cluster center assignment stage (CCA) is applied to reduce the distance computation cost between the points and the cluster centers. The Dunn index [37] is used to measure the quality of the resulting clusters. Fig. 4 provides an overview of QBCA. In this algorithm, the parameter δ is used to determine the condition under which the algorithm terminates, and it is determined as follows:

$$\delta = \frac{1}{k}\sum_{h=1}^{k} \Phi(\mathbf{s}_h^{t-1}, \mathbf{s}_h^t) \tag{17}$$

$$\Phi(\mathbf{s}_h^{t-1}, \mathbf{s}_h^t) = \sum_{l=1}^{m} (s_{h,l}^{t-1} - s_{h,l}^t)^2 \tag{18}$$

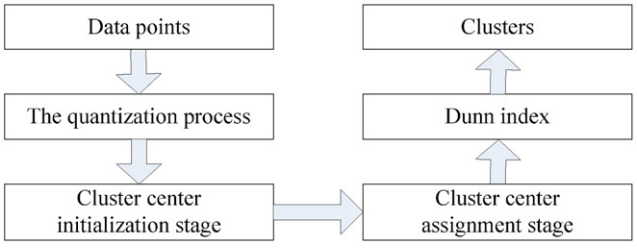


Fig. 3. The framework of the quantization-based clustering algorithm (QBCA).

Algorithm QBCA (Point set P, Seed number k, Threshold ϵ)
1. Perform quantization;
2. Perform CCI;
3. t=0;
4. $\delta = 2\ \epsilon$;
5. Repeat
6. Apply CCA;
7. Recompute the centers of the clusters as the new seeds according to the points assigned to them;
8. t = t +1;
9. Calculate δ;
10. Until $\delta < \epsilon$;

Fig. 4. Quantization-based clustering algorithm.

where k denotes the number of clusters, m denotes the number of dimensions, t denotes the current iteration, $\mathbf{s}_h^t$ with components $s_{h,l}^t$ is the current center of the h-th cluster and $\mathbf{s}_h^{t-1}$ is the center of the h-th cluster in the previous iteration. The convergence of the algorithm is indicated by a small value of δ.

4.1. The quantization process

During the quantization process, QBCA maps all the data points to a m dimensional histogram B. Specifically, we first determine the number of histogram bins per dimension ρ, and the size λ_l of the histogram bin in each dimension by the following equations:

$$\rho = \lfloor \log_m n \rfloor \tag{19}$$

$$\lambda_l = \frac{\overline{p}_l - \underline{p}_l}{\rho} \quad \forall l \in \{1, \ldots, m\} \tag{20}$$

where n is the number of data points, and m is the number of dimensions. The function $\lfloor \cdot \rfloor$ performs a round down operation to the nearest integer, and $\overline{p}_l$ and $\underline{p}_l$ denote the maximum and minimum component value in the l-th dimension, respectively.

Afterwards, all the data points are assigned to their respective histogram bins with sizes $\prod_{l=1}^{m} \lambda_l$. The quantization function is defined as follows:

$$\mathbf{b}.id = \sum_{l \in \{1,\ldots,m\}} (\xi_l - 1)\rho^{m-l} + \xi_m$$

$$\xi_l = \begin{cases} \left\lceil \frac{p_l - \underline{p}_l}{\lambda_l} \right\rceil & \text{if } p_l > \underline{p}_l \\ 1 & \text{if } p_l = \underline{p}_l \end{cases} \tag{21}$$

where $\mathbf{b}.id$ is the index of the histogram bin $\mathbf{b}$, and p_l is the value of the data point $\mathbf{p}$ in the l-th dimension. In other words, the index is determined by linearizing the m-dimensional histogram.

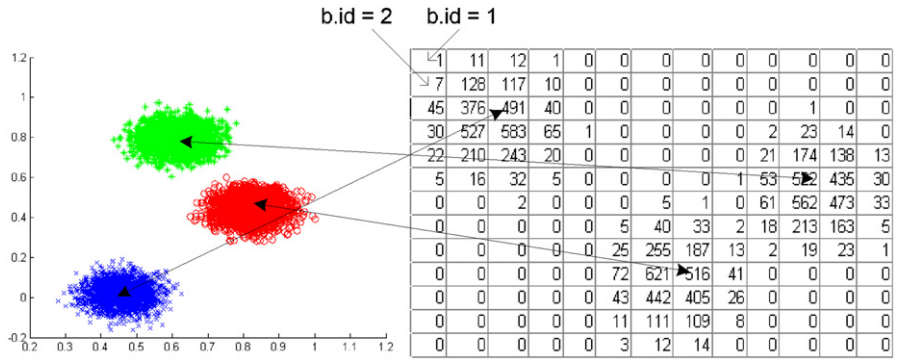
As a result, each data point $\mathbf{p}$ is associated with the corresponding histogram bin $\mathbf{b}$ by the quantization process.

An example data set with 9000 two dimensional data points is shown in Fig. 5(a). This data set can be partitioned into 3 clusters, and each cluster contains 3000 data points. In the first step of the quantization process, the number of histogram bins per dimension ρ is calculated according to Eq. (19), and $\rho = 13$ in this example. Then, the value range of data points in each dimension are divided into 13 equally spaced histogram bins. Finally, all the data points are quantized into their respective histogram bins based on Eq. (21). Fig. 5(b) shows the distribution of the data set among the histogram bins.

4.2. Cluster center initialization stage

The next step of QBCA is to determine the initial centers (seeds) of the clusters. An interesting observation is that the closer a region is to the center of a Gaussian distribution, the higher the number of points in the region will be. Motivated by this property of the Gaussian distribution, we design a cluster center initialization stage (CCI) to select the initial seeds.

CCI (Fig. 6) first (i) initializes an empty Max-heap H (a heap is a specialized tree-based data structure. If the element with the largest value is always at the root node in the heap, the heap is called a max-heap.) and a seed list L, (ii) assigns all the points to their respective histogram bins, (iii) counts the number of points ($|\mathbf{b}_j|$) in each histogram bin $\mathbf{b}_j$ (where $1 \le j \le n_b$, n_b is the number of histogram bins), and (iv) processes all non-zero bins. Then, it performs a set of de-heap operations repeatedly and considers the



1	11	12	1	0	0	0	0	0	0	0	0	0
7	128	117	10	0	0	0	0	0	0	0	0	0
45	376	491	40	0	0	0	0	0	0	1	0	0
30	527	583	65	1	0	0	0	0	2	23	14	0
22	210	243	20	0	0	0	0	0	21	174	138	13
5	16	32	5	0	0	0	0	1	53	512	435	30
0	0	2	0	0	0	5	1	0	61	562	473	33
0	0	0	0	0	5	40	33	2	18	213	163	5
0	0	0	0	0	25	255	187	13	2	19	23	1
0	0	0	0	0	72	621	516	41	0	0	0	0
0	0	0	0	0	43	442	405	26	0	0	0	0
0	0	0	0	0	11	111	109	8	0	0	0	0
0	0	0	0	0	3	12	14	0	0	0	0	0

Fig. 5. An example of the quantization process. (a) Dataset (b) The distribution of data points in histogram bins.

Algorithm CCI (Point set P, Seed number k, Threshold ϵ)
1. Initialize an empty Max-heap H and a seed list L;
2. Assign all the points to their respective histogram bins;
3. Calculate the number of points ($|b_j|$) in each histogram bin b_j;
   ($1 \le j \le n_b$, n_b is the number of histogram bins)
4. Process all non-zero bins;
5. While (H is not empty)
6. get the next bin b of H;
7. flag = true;
8. For each neighboring bin b_i of b
9. If ($|b_i| > |b|$) flag = false;
10. If (flag) add b into L;
11. If ($|L| < k$)
12. Select $k - |L|$ histogram bins with the largest cardinalities from the remaining histogram bins which are not in the list L and add them into L
13. For the first k histogram bins with the largest cardinalities in L
14. Cluster all the points in each histogram bin as one cluster;
15. The center of the cluster is viewed as a seed;

Fig. 6. The cluster center initialization stage (CCI).

neighboring histogram bin $\mathbf{b}_i$ of the de-heaped entry $\mathbf{b}$ one by one (where the de-heap operation means removing the root node of a heap). If none of the cardinalities of the neighboring bins are greater than that of the bin $\mathbf{b}$, $\mathbf{b}$ is added to the seed list L. The loop terminates when the heap is empty or the number of histogram bins $|L|$ in L is not smaller than k. If $|L| < k$, the list L adds $k-|L|$ histogram bins with the largest cardinalities from the remaining histogram bins which are not in the list L. Afterwards, CCI considers the first k bins $\mathbf{b}_j$ with the largest cardinalities in L as follows: (i) it clusters all the points in $\mathbf{b}_j$ as one cluster and (ii) the center of the cluster is viewed as a seed. Finally, CCI obtains a list of seeds (the initial centers of clusters).

Fig. 7(a) provides an illustration of this stage for the case $k=2$. CCI: (i) assigns all the black points to their respective histogram bins and (ii) inserts all the histogram bins $\mathbf{b}_j$ with cardinality $|\mathbf{b}_j|$ into the Max-heap H. Then, it performs the de-heap operation repeatedly. The first de-heaped entry is $\mathbf{b}_2$ with cardinality 7 (the gray histogram bin with slashes in Fig. 7(a)). CCI considers the neighboring histogram bins $\mathbf{b}_1$, $\mathbf{b}_3$ one by one. Since the cardinalities of all the neighbors are smaller than that of the bin $\mathbf{b}_2$, the bin $\mathbf{b}_2$ is added to the seed list L. The next de-heaped entry is the bin $\mathbf{b}_8$. The seed list L is then updated by the bin $\mathbf{b}_8$ since its cardinality is greater than that of its neighbors. The loop terminates when the cardinality of the seed list L is equal to k, and L contains the histogram bins $\{\mathbf{b}_2, \mathbf{b}_8\}$. The points in the

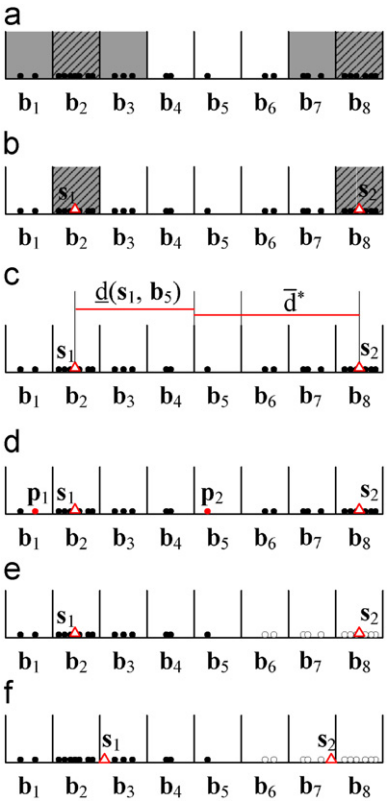


Fig. 7. An example to illustrate QBCA.

histogram bins $\{\mathbf{b}_2, \mathbf{b}_8\}$ are clustered and represented by the seed points as shown in Fig. 7(b). Finally, CCI obtains two seeds $\mathbf{s}_1$ and $\mathbf{s}_2$ (the red triangles in Fig. 7(b)).

4.3. Cluster center assignment stage

The next step of QBCA is to obtain the seed candidate lists of the non-empty histogram bins, and assign the points in the bins to the closest seed. The motivation of our currently proposed cluster center assignment stage (CCA) is based on the properties of the seed candidate list which are summarized in Lemmas 1 and 2.

CCA (Fig. 8) considers the non-empty histogram bins $\mathbf{b}_j$ one by one. It first (i) calculates the maximum distance between the histogram bin $\mathbf{b}_j$ and the seeds, (ii) sorts the maximum distances in ascending order, and (iii) selects the first distance value and denote it as $\overline{d}^*(\mathbf{s},\mathbf{b})$. Then, it computes $\underline{d}(\mathbf{s}_i,\mathbf{b}_j)$ for each seed $\mathbf{s}_i$

Algorithm CCA (Point set P, Histogram B, Seed number k)
1. For each non-empty histogram bin b_j in B
2. Calculate the maximum distance $\overline{d}$ between the bin b_j and the seeds;
3. Sort the maximum distances in ascending order;
4. Select the first distance value as $\overline{d}^*(s, b)$;
5. For each seed s_h ($s_h \in S$) (where S is a seed set);
6. If $\underline{d}(s_h, b_j) \le \overline{d}^*(s, b)$
7. add s_h to the seed candidate list $b_j.list$ of the bin b_j;
8. For each point p in the histogram bin b_j
9. Calculate the distance between the point p and the seeds in the seed candidate list $b_j.list$;
10. Assign p to the seed candidate which is closest to it;

Fig. 8. The cluster center assignment stage (CCA).

(where $\underline{d}(\mathbf{s}_i,\mathbf{b}_j)$ denotes the minimum distance between the seed $\mathbf{s}_i$ and the bin $\mathbf{b}_j$). If $\underline{d}(\mathbf{s}_i,\mathbf{b}_j) \le \overline{d}^*(\mathbf{s},\mathbf{b})$, CCA adds $\mathbf{s}_i$ to $\mathbf{b}_j.list$ as the seed candidates of $\mathbf{b}_j$ (where $\mathbf{b}_j.list$ stores the seed candidates for the histogram bin $\mathbf{b}_j$). The loop terminates when all the non-empty histogram bins are considered. Afterwards, CCA considers the point $\mathbf{p}$ in each histogram bin one by one. If the histogram bin $\mathbf{b}$ only contains one seed candidate $\mathbf{s}$, the point $\mathbf{p}$ which belongs to the histogram $\mathbf{b}$ can be assigned to the seed $\mathbf{s}$ directly without the distance computation. Otherwise, it calculates the distance between the point $\mathbf{p}$ and the seeds in the seed candidate list $\mathbf{b}_j.list$. Then, it assigns the point $\mathbf{p}$ to the seed which is closest to it. CCA terminates when all the points are assigned to the nearest seeds.

Fig. 7(c) and (d) provide an example of CCA. CCA considers the non-empty histogram bins one by one. The first histogram bin is the bin $\mathbf{b}_1$ as shown in Fig. 7(c). $\overline{d}^*(\mathbf{s},\mathbf{b})$ is equal to the maximum distance $\overline{d}(\mathbf{s}_1,\mathbf{b}_1)$ between the seed $\mathbf{s}_1$ and the bin $\mathbf{b}_1$. We then calculate $\underline{d}(\mathbf{s}_1,\mathbf{b}_1)$ and $\underline{d}(\mathbf{s}_2,\mathbf{b}_1)$. Since $\underline{d}(\mathbf{s}_1,\mathbf{b}_1) < \overline{d}^*(\mathbf{s},\mathbf{b})$ and $\underline{d}(\mathbf{s}_2,\mathbf{b}_1) > \overline{d}^*(\mathbf{s},\mathbf{b})$, $\mathbf{s}_1$ is added to $\mathbf{b}_1.list$. On the other hand, for the histogram bin $\mathbf{b}_5$, $\overline{d}^*(\mathbf{s},\mathbf{b})$ is equal to $\overline{d}(\mathbf{s}_2,\mathbf{b}_5)$, which is the maximum distance between the seed $\mathbf{s}_2$ and the histogram bin $\mathbf{b}_5$. Since $\underline{d}(\mathbf{s}_1,\mathbf{b}_5) < \overline{d}^*(\mathbf{b},\mathbf{s})$ and $\underline{d}(\mathbf{s}_2,\mathbf{b}_5) < \overline{d}^*(\mathbf{s},\mathbf{b})$, the seed candidate list $\mathbf{b}_5.list$ contains the seeds $\mathbf{s}_1$ and $\mathbf{s}_2$. After obtaining the seed candidate list for each bin, CCA assigns the points in the bins to the closest seeds by only considering the seed candidates in the seed candidate list. The point $\mathbf{p}_1$, which belongs to the bin $\mathbf{b}_1$ in Fig. 7(d), is assigned to the seed $\mathbf{s}_1$ directly since the seed candidate list $\mathbf{b}_1.list$ only contains the seed $\mathbf{s}_1$. The point $\mathbf{p}_2$, which belongs to the bin $\mathbf{b}_5$ in Fig. 7(d), is assigned to the seed $\mathbf{s}_1$ since $d(\mathbf{s}_1,\mathbf{p}_2) < d(\mathbf{s}_2,\mathbf{p}_2)$ (where $d(\cdot,\cdot)$ is the Euclidean distance, and $\mathbf{s}_1$, $\mathbf{s}_2$ are in the seed candidate list of the bin $\mathbf{b}_5$).

4.4. Re-computation of the cluster centers

After assigning all the points to the seeds, the algorithm performs the following steps: (i) it recomputes the centers of the clusters and uses these centers as the new seeds and (ii) calculates the gap δ by Eqs. (17) and (18). The algorithm terminates when $\delta < \varepsilon$, where ε is a suitably chosen threshold. Fig. 7(e) shows the clusters before the re-computation of the centers, while Fig. 7(f) illustrates the clusters with their centers after re-computation. Although we have only illustrated a single iteration of the algorithm, it can be observed that the convergence of the algorithm is rapid by comparing Fig. 7(e) with (f).

4.5. Performance analysis

QBCA is mainly composed of three stages: the quantization stage, the cluster center initialization stage (CCI) and the cluster center assignment stage (CCA). The time consumption T_{QBCA} of QBCA is affected by the time of the quantization stage T_{Quant}, the time of cluster center initialization stage T_{CCI}, the time for cluster center assignment stage T_{CCA}, and the number of iterations α as follows:

$$T_{QBCA} = T_{Quant} + T_{CCI} + \alpha \cdot T_{CCA} \tag{22}$$

The computational cost of the stages in QBCA is related to the number of data points n, the number of histogram bins $|B|$, the number of clusters k, the number of dimensions m, the average number of seed candidates η in each histogram bin $\mathbf{b}$, and the number of iterations α. Specifically, QBCA first assigns all the data points to their respective bins in the histogram with a time complexity of $O(n)$. Then, it processes all the non-zero bins using the heap in CCI with time complexity $O(2^m \cdot |B| \log |B|)$. Next, QBCA calculates seed candidates for all the non-zero bins with time complexity $O(\alpha k |B|)$, and assigns data points to the corresponding clusters with time complexity $O(\alpha \eta n)$. As a result, the time complexity of QBCA is $O(n + 2^m \cdot |B| \log |B| + \alpha \cdot (k \cdot |B| + \eta \cdot n))$. In the current special case, the functional dependence of $|B|$ on m and n in the complexity estimation is $\lfloor \log_m n \rfloor^m$, and this is much smaller than n when n is large and m is small. For example, when $n = 1\,000\,000$, $m = 3$, $\lfloor \log_m n \rfloor^m$ is equal to 1728, which means that the assumption of $|B| \ll n$ is valid. In addition, since k, η, α are much smaller than n, and 2^m is small when m is small, the time complexity of QBCA is approximately $O(n)$.

The space consumption consists of two components: (i) the storage of the data points, and (ii) the storage of the ids and the cardinalities of the histogram bins. Since the number of histogram bins is smaller than the number of data points, the space complexity of QBCA is $O(n)$.

4.6. Cluster quality measure

We adopt the Dunn index [37] to choose the number of clusters k and to measure the quality of the resulting clusters. As a function of k, the larger the value of the Dunn index, the better the corresponding k value will be for characterizing the underlying cluster structure. Given a set of cluster results $\{R_1, \ldots, R_k\}$ obtained by QBCA, the Dunn index (Δ_k) is defined as follows:

$$\Delta_k = \min_{i=1,\ldots,k} \left\{ \min_{h=i+1,\ldots,k} \left(\frac{d(R_i,R_h)}{\max_{j=1,\ldots,k} d_{diam}(R_j)} \right) \right\} \tag{23}$$

$$d(R_i,R_h) = \min_{\mathbf{x} \in R_i, \mathbf{y} \in R_h} d(\mathbf{x},\mathbf{y}) \tag{24}$$

$$d_{diam}(R_j) = \max_{\mathbf{x},\mathbf{y} \in R_j} d(\mathbf{x},\mathbf{y}) \tag{25}$$

where $d(\mathbf{x}, \mathbf{y})$ is the Euclidean distance between two points $\mathbf{x}$ and $\mathbf{y}$, d_{diam} is the diameter of a cluster, and $d(R_i, R_h)$ measures the dissimilarity between two regions R_i and R_h. The best clustering result corresponds to a large $d(R_i, R_h)$ and a small $d_{diam}(R_j)$.

The best k value is determined as follows:

$$k^* = \arg \max_{k_{\min} \le k \le k_{max}} \Delta_k \tag{26}$$

where k_{min} and k_{max} are the minimum and maximum values in our adopted range of k, respectively. Clustering is then performed based on this optimum k value.

Compared with K-means, QBCA achieves a low computation time without sacrificing the quality of the clusters, which represents one of its main advantages.

5. Shrinking process

As mentioned above, the performance of QBCA is sensitive to the average number of seed candidates η in each histogram bin. In

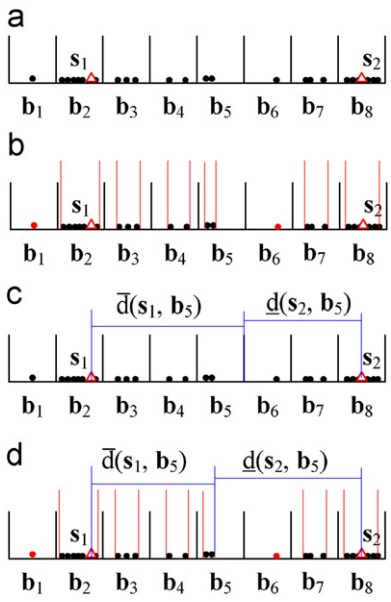


Fig. 9. An example of the shrinking process. (a) Before shrinking; (b) after shrinking;(c) candidates before shrinking; (d) candidates after shrinking.

Algorithm Shrinking (Point set P, Histogram B)
1. For each bin b (b $\in B$)
2. $\underline{b'}_l = \infty$, $\overline{b'}_l = -\infty$, $(1 \le l \le m)$;
3. For each point p (p $\in P$) and its corresponding histogram bin b;
4. For each dimension l;
5. If $(p_l < \underline{b'}_l)$
6. $\underline{b'}_l = p_l$;
7. If $(p_l > \overline{b'}_l)$
8. $\overline{b'}_l = p_l$;

Fig. 10. Shrinking process.

order to improve the performance of QBCA further, we reduce the number of seed candidates in each histogram bin by a shrinking process.

Fig. 9(a) and (b) illustrate the shrinking process by an example in a set of one dimensional histogram bins. It can be observed that (i) If a histogram bin **b** only contains one point **p**, the boundary of **b** is represented by the point **p**. (ii) If the histogram bin **b** contains more than one point, the boundary of the histogram bin **b** is shrunken to the histogram bin **b**′ which is determined by the following equations:

$$\underline{b'}_l = \min_{\forall \mathbf{p} \in \mathbf{b}} p_l \quad \text{for } 1 \le l \le m \tag{27}$$

$$\overline{b'}_l = \max_{\forall \mathbf{p} \in \mathbf{b}} p_l \quad \text{for } 1 \le l \le m \tag{28}$$

where $\underline{b'}_l$ and $\overline{b'}_l$ denote the corresponding minimum and maximum component value of the histogram bin **b**′ in the l-th dimension, **p** is the point which belongs to the histogram bin **b**, p_l is the l-th component value of the point **p**, and m is the number of dimensions.

Specifically, the shrinking process (Fig. 10) first (i) initializes the minimum component value $\underline{b'}_l$ and the maximum component value $\overline{b'}_l$ of the histogram bin **b**′ (which is the shrunken version of the boundary of the histogram bin **b**) in each dimension and (ii) considers the points inside the corresponding histogram bin **b** one by one. If $p_l < \underline{b'}_l$, the component value $\underline{b'}_l$ is replaced by p_l. If $p_l > \overline{b'}_l$, $\overline{b'}_l$ is replaced by p_l. The shrinking process terminates when all the points are considered.

Fig. 9(c) and (d) provide an example to illustrate the effect of the shrinking process. The histogram bin $\mathbf{b}_5$ has two seed candidates $\{\mathbf{s}_1, \mathbf{s}_2\}$ before shrinking as shown in Fig. 9 (c), since $\overline{d}^*(\mathbf{s},\mathbf{b}) = \overline{d}(\mathbf{s}_1,\mathbf{b}_5)$, $\underline{d}(\mathbf{s}_1,\mathbf{b}_5) < \overline{d}^*(\mathbf{s},\mathbf{b})$ and $\underline{d}(\mathbf{s}_2,\mathbf{b}_5) < \overline{d}^*(\mathbf{s},\mathbf{b})$. After shrinking as shown in Fig. 9 (d), $\overline{d}^*(\mathbf{s},\mathbf{b}) = \overline{d}(\mathbf{s}_1,\mathbf{b}_5)$ is reduced when compared with the original $\overline{d}^*(\mathbf{s},\mathbf{b})$, while $\underline{d}(\mathbf{s}_2,\mathbf{b}_5)$ increases. As a result, $\underline{d}(\mathbf{s}_2,\mathbf{b}_5) > \overline{d}^*(\mathbf{s},\mathbf{b})$. The seed candidate list of the histogram bin $\mathbf{b}_5$ now only contains $\mathbf{s}_1$.

6. Hierarchical structure

In QBCA, we calculate the distances of all the non-empty histogram bins to the k centers of the clusters. Fig. 11(a) shows an example of QBCA based on a single level histogram when $k=2$ in one dimensional space (here we assume that all the histogram bins are non-empty). For example, in order to obtain the seed candidates, we compute the minimum and maximum distance of the histogram bins $\{\mathbf{b}_3, \mathbf{b}_4\}$ to all the seeds $\{\mathbf{s}_1, \mathbf{s}_2\}$. The minimum and maximum distances of the rest of the bins $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_4, \mathbf{b}_5, \mathbf{b}_6, \mathbf{b}_7$ and $\mathbf{b}_8\}$ are calculated in a similar way. Fig. 11(b) shows another example of QBCA based on a hierarchical histogram when $k=2$. In this case we only need to calculate the minimum and maximum distance of this single parent histogram bin $\mathbf{b}^{parent}$, which contains the child histogram bins $\{\mathbf{b}_3, \mathbf{b}_4\}$, to these seeds. An interesting observation is that the child histogram bins can obtain their seed candidates from the seed candidate list $\mathbf{b}^{parent}.list$ of the parent histogram bin $\mathbf{b}^{parent}$. According to properties of the maximum and minimum distance values and the relationship between the parent histogram bin and the child histogram bins, the following lemma is obtained.

Lemma 3. *If* $\mathbf{b} \subseteq \mathbf{b}^{parent}$, *the seed candidates of the child histogram bin* **b** *is a subset of the seed candidates of the parent histogram bin* $\mathbf{b}^{parent}$:

$$\mathbf{b}.list \subseteq \mathbf{b}^{parent}.list \tag{29}$$

where **b**.*list and* $\mathbf{b}^{parent}$.*list denote the seed candidate list of the child histogram bin and the parent histogram bin, respectively*.

The proof of Lemma 3 is given in Appendix C.

We now provide a more detailed explanation of the construction, selection, and assignment of child bins to parent bins based on Fig. 11 (In the figure, two seeds $\{\mathbf{s}_1,\mathbf{s}_2\}$ are denoted as small red triangles, and the boundaries of the parent histogram bins $\mathbf{b}^{parent}$ are represented by long red lines). QBCA first subdivides the value range space into four equally spaced histogram bins $\{\mathbf{b}_1^{parent}$, $\mathbf{b}_2^{parent}$, $\mathbf{b}_3^{parent}$, $\mathbf{b}_4^{parent}\}$. Then, each parent histogram bin $\mathbf{b}^{parent}$ is associated with a set of the seed candidates. For example, the seed candidate set of $\mathbf{b}_2^{parent}$ is $\{\mathbf{s}_1\}$, while the seed candidate set of $\mathbf{b}_3^{parent}$ is $\{\mathbf{s}_1,\mathbf{s}_2\}$. Next, the parent histogram bin is further divided into two equally spaced child histogram bins. According to

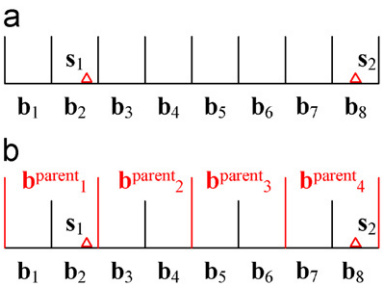


Fig. 11. An example of the hierarchical histogram structure. (a) Original histogram; (b) hiararchical histogram.

Lemma 3, the seed candidates of the two child histogram bins can be selected from the set of seed candidates in the parent histogram bins. For example, the seed candidate set of the two child histogram bins $\mathbf{b}_3$ and $\mathbf{b}_4$ is $\{\mathbf{s}_1\}$, while the seed candidate set of the two child histogram bins $\mathbf{b}_5$ and $\mathbf{b}_6$ is $\{\mathbf{s}_1,\mathbf{s}_2\}$.

According to the property of this hierarchical structure, the following corollary is obtained.

Corollary 1 (*Hierarchical relationship of the candidate list*). *Let* $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_E\}$ *be a set of histogram bins and* $\{\mathbf{b}_1.list, \mathbf{b}_2.list, \ldots, \mathbf{b}_E.list\}$ *be the corresponding seed candidate lists of the histogram bins* (*where E is the total number of histogram bins in the set*). *If the histogram bins satisfy the hierarchical relationship* $\mathbf{b}_1 \supseteq \mathbf{b}_2 \supseteq \cdots \supseteq \mathbf{b}_E$, *the seed candidate lists of the histogram bins satisfy the hierarchical relationship* $\mathbf{b}_1.list \supseteq \mathbf{b}_2.list \supseteq \cdots \supseteq \mathbf{b}_E.list$.

Corollary 1 describes the hierarchical relationship between the histogram bins, and its proof is given in Appendix D.

The time complexity of constructing a hierarchical structure which indexes the histogram bins is $O(|B|\log|B|)$ (where $|B|$ is the number of histogram bins). Note that the hierarchical structure only indexes the histogram bins, not the data points.

If the shrinking process is performed on the hierarchical structure, the bins in the hierarchical structure satisfy the following conditions:

$$\underline{b}_l^e = \min_{\forall \mathbf{b}^{e+1} \in \mathbf{b}^e} \underline{b}_l^{e+1} \quad \forall l \in \{1,\ldots,m\} \tag{30}$$

$$\overline{b}_l^e = \max_{\forall \mathbf{b}^{e+1} \in \mathbf{b}^e} \overline{b}_l^{e+1} \quad \forall l \in \{1,\ldots,m\} \tag{31}$$

where $\mathbf{b}^e$ and $\mathbf{b}^{e+1}$ denote the bins at the e th level and $(e+1)$th level, respectively, the bin $\mathbf{b}^{e+1}$ is the child of the bin $\mathbf{b}^e$, and $\underline{b}_l^e$, $\overline{b}_l^e$ ($\underline{b}_l^{e+1}$, $\overline{b}_l^{e+1}$) denote the minimum and maximum values of the bins $\mathbf{b}^e$ ($\mathbf{b}^{e+1}$) in the l-th dimension, respectively.

Based on the property of the hierarchical structure and the shrinking process, the following lemma is obtained.

Lemma 4. *The hierarchical relation of the candidate list is still satisfied after applying the shrinking process to the hierarchical structure.*

The proof of Lemma 4 is given in Appendix E.

According to Lemma 4, the shrinking process reduces the number of candidates for each bin at different levels of the hierarchical structure. As a result, QBCA based on the hierarchical structure achieves good performance through the shrinking process.

7. Experiment

7.1. Experimental setting and data sets

We experimentally evaluate the efficiency of the proposed approach in this section. First, two synthetic data sets (10 000 points) in the hypercube $[0, 1]^3$ are generated as follows: A set of cluster centers are sampled from a uniform distribution in the hypercube, and the data points are generated using a Gaussian distribution with these centers as the mean vectors. Specifically, two data sets "Gaussian (5)" and "Gaussian (25)", which are shown in Fig. 12(a) and (b), are synthesized with 5 and 25 Gaussian clusters, with corresponding standard deviations 0.06 and 0.02.

Then, QBCA is applied to perform segmentation on an image collection. The images are collected from the databases in [32] and from the web. The sizes of images in the collection vary from 128×96 to 2448×1632. Each pixel of the image is associated with a feature vector consisting of the L, a, and b components of the CIE L*a*b* color space. Hence, each pixel is a point in $\mathbb{R}^3$ and the similarity between two pixels is measured by the Euclidean distance.

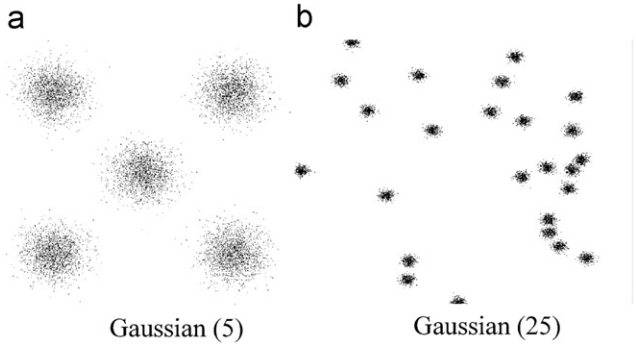


Fig. 12. Data sets.

We also apply the following Gaussian filter $G(w_1,w_2)$ to perform smoothing on the image prior to segmentation:

$$G(w_1,w_2) = \frac{G_g(w_1,w_2)}{\sum_{w_1}\sum_{w_2} G_g} \tag{32}$$

$$G_g(w_1,w_2) = e^{-\frac{(w_1^2+w_2^2)}{2\sigma^2}} \tag{33}$$

where $G(w_1, w_2)$ denotes a rotationally symmetric Gaussian low pass filter of size 5×5 with standard deviation σ. In our experiments, we choose $\sigma = 0.5$.

Finally, three data sets (the character trajectories data set, the iris data set and the wine data set) in the UCI machine learning repository [49] are used to study the performance of QBCA, QBCA without the hierarchical structure (QBCA(-H)) and QBCA without the hierarchical structure and the shrinking process (QBCA(-HS)) in terms of the accuracy and the efficiency. The character trajectories data set consists of 2858 character samples, which can be categorized into 20 classes. Each character sample is represented by a 3-dimensional pen tip average velocity in the trajectory. Fig. 13(a) shows the distribution of pen tip average velocities in the first five classes in the character trajectories data set. The iris data set contains 3 classes, and each class includes 50 samples. Fig. 13(b) illustrates the distribution of the iris data set after projection by Principle Component Analysis (PCA) on to a three dimensional space. The wine data set contains 178 samples which are the results of a chemical analysis of three types of wines. Fig. 13(c) shows the distribution of the wine data set after projection on to a three dimensional space.

In the following experiments, we apply QBCA and variants of K-means based on kd-tree [10], including the swap algorithm (SWAP), the simple hybrid algorithm (EZ-Hybrid), and the Hybrid algorithm (Hybrid) on the synthetic data set and the image collection.

The performance of QBCA is evaluated through the following measures: (i) the computation time, (ii) the average number of distance computations per iteration, and (iii) the average distortion of clusters. The measures (i) and (ii) can be used to determine the efficiency of QBCA. The smaller the values of these measures, the faster the algorithm will become. The average distortion of clusters $\overline{\phi}$ measures the effectiveness of QBCA, which is defined as follows:

$$\overline{\phi} = \frac{1}{n}\sum_{h=1}^{k}\sum_{i=1}^{|S_h|}(\mathbf{p}_i - \mathbf{s}_h)^2 \tag{34}$$

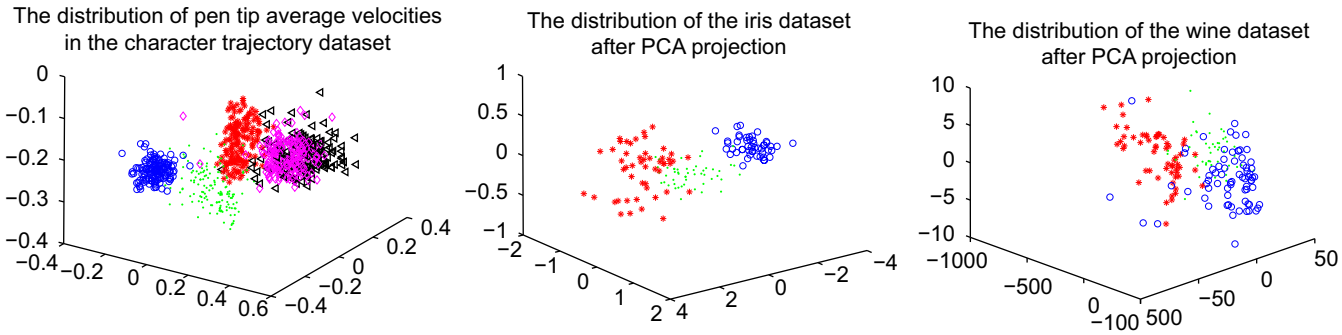


Fig. 13. The distribution of the different data sets: (a) the character trajectories data set; (b) the iris data set; and (c) the wine data set.

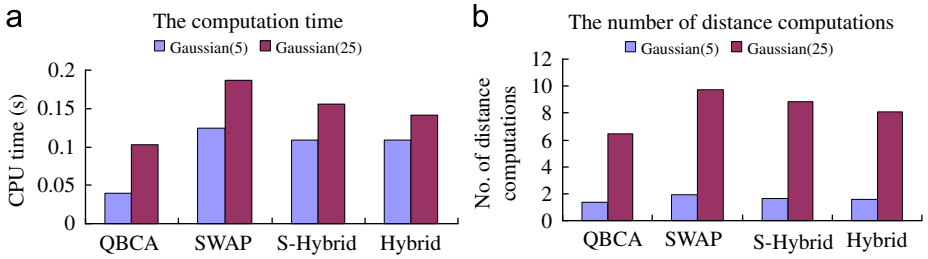


Fig. 14. Comparison in terms of efficiency: (a) the computation time and (b) the average number of distance computations.

where $\mathbf{s}_h$ is the center of the cluster S_h, and $|S_h|$ is the cardinality of the cluster S_h.

7.2. Comparison on synthetic data set

In order to evaluate the performance of the algorithm, we first compare these algorithms on two synthetic data sets. The value of k is set to 5 and 25 for the synthetic data sets Gaussian (5) and Gaussian (25), respectively. The termination threshold ε of all the algorithms is set to 0.0001. For the other three algorithms, we adopt the implementation in [10]. The maximum number of points stored in each leaf node of the kd-tree for all these algorithms is 50, which serves as a good compromise between the two extreme cases of either including a large number of points in a leaf node, or using a large number of nodes in the tree. The maximum number of iterations for all the algorithms is set to 30, which serves as a good trade off between accuracy and efficiency: While a large number of iterations may improve the accuracy, the resulting computation time will also be long. On the other hand, the number of centers adopted will be different for different data sets. In addition, for the SWAP algorithm, the maximum number of swaps at each iteration is set to 1, since it is observed that the results are satisfactory with this setting while the computation time is minimized.

It can be observed that the quality of all the algorithms, which is measured by the average distortion of clusters as shown in Table 2, are comparable, expect for the SWAP algorithm, while QBCA results in a significant reduction in computation time as shown in Fig. 14(a). There are two reasons which lead to QBCA outperforming its competitors. First, the initialization time of QBCA is lower than that of the other algorithms as shown in Table 1, since the time complexity of assigning the points to their respective histogram bins is $O(n)$, while the time complexity of constructing a kd-tree is $O(n \log n)$. Second, QBCA achieves low computation time by reducing the number of distance computations for each point. In particular, we apply CCI to determine the initial centers of the clusters, and although the positions of these initial cluster center positions are only approximate, they are already very close to the actual positions. As a result, QBCA converges quickly. In addition, we have applied the cluster center assignment stage to reduce the number of distance computations for each point. While we need to calculate the distances between the current point and the k cluster centers in K-means, we only need to compute the distances between the same point and η seed candidates in the seed candidate list of the bins (where $\eta \ll k$) in QBCA. As a result, the average number of distance computations in QBCA is significantly smaller than those of other algorithms as shown in Fig. 14(b).

Table 1
The initialization time.

Methods	Gaussian (5)	Gaussian (25)
QBCA	0.00396	0.00403
Swap	0.0621	0.0624
EZ-Hybrid	0.0621	0.0624
Hybrid	0.0621	0.0624

Table 2
Average distortion of the clusters.

Methods	Gaussian (5)	Gaussian (25)
QBCA	0.007487	0.00227
Swap	0.0103	0.0167
EZ-Hybrid	0.007489	0.00231
Hybrid	0.007488	0.00229

7.3. Comparison of segmentation efficiency on the image collection

In order to further compare the performance of the algorithms, we apply these algorithms to perform image segmentation on 6 images with different sizes which are randomly selected from the image collection. Fig. 15 shows the selected images with their

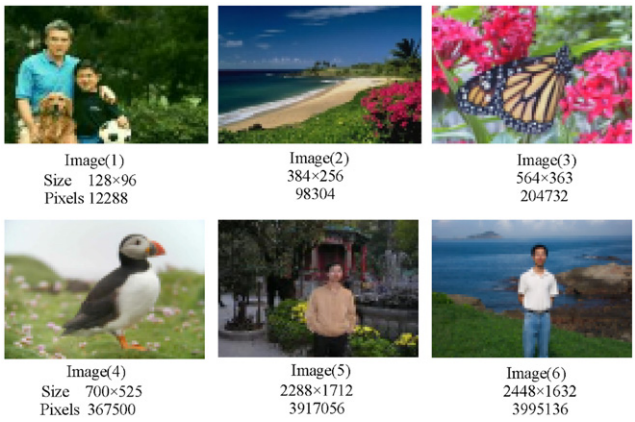


Fig. 15. Selected images for evaluating segmentation efficiency.

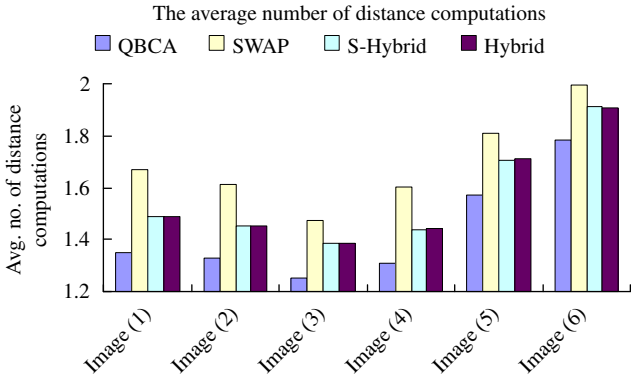


Fig. 16. The average number of distance computations.

Table 3
The computation time (in seconds).

Images	QBCA	Swap	EZ-Hybrid	Hybrid
Image (1)	0.065	0.234	0.25	0.25
Image (2)	0.291	1.391	1.297	1.344
Image (3)	0.766	3.297	3.125	3.157
Image (4)	1.219	3.985	3.844	3.766
Image (5)	19.262	112.23	103.05	105.12
Image (6)	19.621	125.96	113.58	114.26

Table 4
The initialization time (in seconds).

Images	QBCA	Swap	EZ-Hybrid	Hybrid
Image (1)	0.004	0.078	0.078	0.078
Image (2)	0.032	0.719	0.719	0.719
Image (3)	0.078	1.578	1.578	1.578
Image (4)	0.141	2.797	2.797	2.797
Image (5)	11.641	81.95	81.95	81.95
Image (6)	11.657	88.32	88.32	88.32

Table 5
Average distortion of the clusters.

Images	QBCA	Swap	EZ-Hybrid	Hybrid
Image (1)	962.56	1354.62	969.89	1009.38
Image (2)	1551.53	2094.15	1558.65	1555.04
Image (3)	2375.1	3332.16	2378.1	2381.7
Image (4)	538.55	751.75	545.15	589.39
Image (5)	568.39	705.31	570.21	568.78
Image (6)	1006.49	1310.02	1008.65	1008.14

corresponding sizes and number of pixels. The termination threshold ε of all the algorithms is set to 0.01, the maximum number of iterations for all the algorithms is set to 50, while the k value is set to 5. Note that, to compare the efficiency of the algorithms, we adopt this particular k value for image segmentation, which may not be the optimal value for the image.

QBCA outperforms other algorithms again for all the images as shown in Table 3, in particular when the sizes of the images are large. At the same time, the minimum average distortion of the clusters is comparable to those of other algorithms, as shown in Table 5. The initialization times of other algorithms based on kd-tree are significantly greater than that of QBCA as shown in Table 4. While the objective of kd-tree is to reduce the average number of distance computations between the points and the cluster centers, when the number of pixels increases, the computation time for building the kd-tree is still significant. On the other hand, the initialization time of QBCA is small even when the number of pixels is large, due to the adoption of the quantization process. In addition, QBCA can also reduce the average number of distance computations as shown in Fig. 16. The corresponding image segmentation results are shown in Fig. 17, and we can observe that the results based on different algorithms are essentially indistinguishable, while QBCA can achieve a low computation time.

We also investigate the relative contributions of CCI and CCA in QBCA. As shown in Fig. 18, the average number of seed candidates in each histogram bin is 5, before performing CCI and CCA. After performing CCI, the average number of seed candidates reduces significantly to a value below 2. The possible reason is that the initial cluster center positions determined by CCI are close to the true center positions. After performing CCA, the average number of seed candidates in each histogram bin decreases further to a value below 1.5. This indicates that most of the histogram bins contain only a single seed candidate. As a result, both CCI and CCA stages are essential to the success of QBCA.

We further compare QBCA with other clustering algorithms, such as the General C-Means approach (GCM) [38] and the Mean Shift (MS) algorithm [39,40] on our image collection. The number of clusters is set to 2 in this experiment, and we use a value of 2 for the distance measure parameter m in GCM, as described in [38]. For the MS algorithm, the kernel bandwidth parameters h_s, h_r associated with the spatial domain and the range domain are set to 16 and 20, respectively, while the threshold M, which eliminates spatial regions containing less than M pixels, is set to 100. We apply QBCA, GCM and MS to perform image segmentation on two randomly selected images as shown in Fig. 19. The computation times obtained using QBCA on both images are significantly lower than those obtained using GCM and MS, respectively, as shown in Table 6, while the corresponding segmentation results obtained using QBCA are very similar with those of GCM and MS as shown in Fig. 20. The possible reasons are: (1) the initial centers of the clusters obtained using CCI are very close to the actual positions and (2) the CCA stage reduces the distance computation cost between the points and the centers of the clusters.

7.4. The effect of the shrinking process and the hierarchical structure

We further study the performance of QBCA, QBCA without the hierarchical structure (QBCA(-H)) and QBCA without the hierarchical structure and the shrinking process (QBCA(-HS)) based on

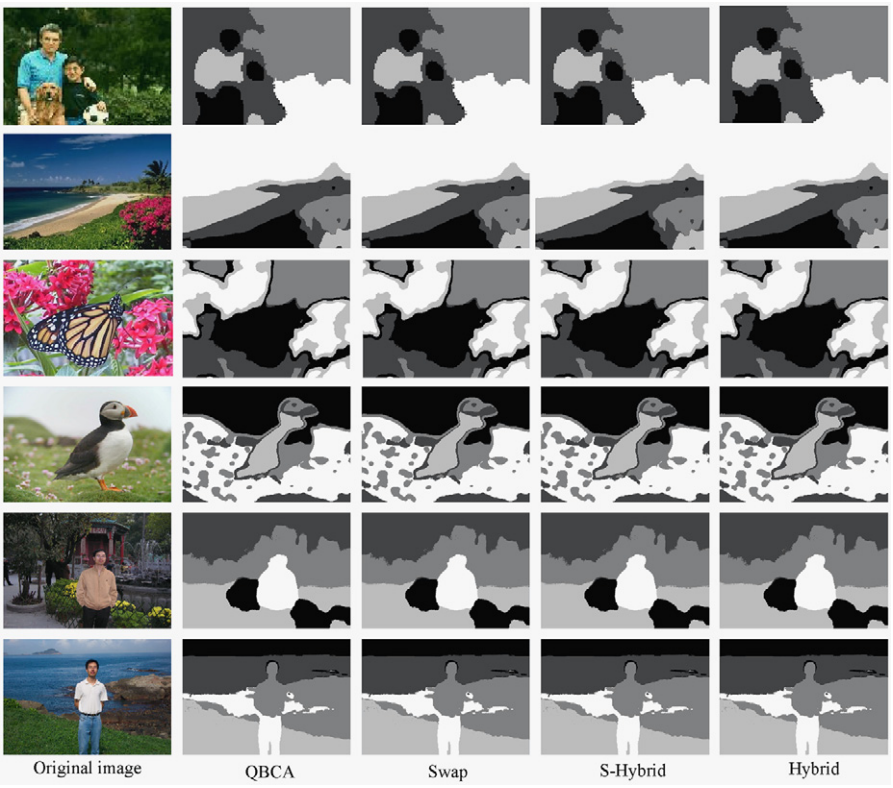


Fig. 17. Image segmentation.

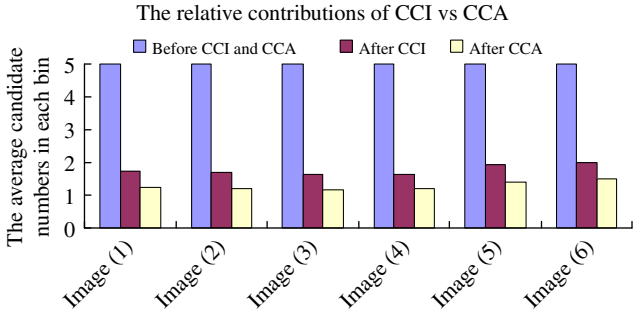


Fig. 18. The investigation of the relative contributions of CCI and CCA in QBCA.

Fig. 19. Selected images for evaluating the performance of QBCA, GCM and MS.

Table 6
Comparison of QBCA, GCM and MS with respect to the computation time (in seconds).

Images	QBCA	GCM	MS
Image (7)	0.276	3.72	2.88
Image (8)	0.287	3.76	2.95

three data sets, which are the character trajectories data set, the iris data set and the wine data set in the UCI machine learning repository[49].

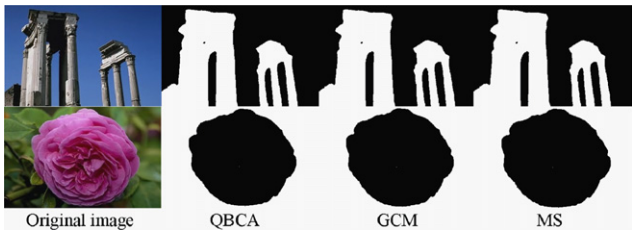


Fig. 20. Segmentation results obtained by QBCA, GCM and MS.

Table 7
Comparison of QBCA, QBCA(-H) and QBCA(-HS) with respect to the accuracy (RI).

Data sets	QBCA	QBCA(-H)	QBCA(-HS)
The character trajectories data set	0.9370	0.9367	0.9363
The iris data set	0.9436	0.9434	0.9427
The wine data set	0.7187	0.7187	0.7185

The accuracy of QBCA is evaluated by the average value of the Rand index [50] after performing QBCA 10 times. Given the ground truth partition Y and the partition Y' obtained by QBCA, we can construct an $n \times n$ adjacency matrix $\mathbf{M}$ with entries m_{ij} ($i,j \in \{1,\ldots,n\}$) for a data set P with n data points. The entry $m_{ij} \in \{1,\ldots,4\}$ is defined as follows: (1) If two data points $\mathbf{p}_i$ and $\mathbf{p}_j$ belong to the same cluster in both partitions Y and Y' or belong to different clusters in both partitions Y and Y', respectively, $m_{ij}=1$. (2) If two data points belong to the same cluster in the partition Y and belong to different clusters in the partition Y', or belong to different clusters in the partition Y but the same cluster in the partition Y', $m_{ij}=2$. Let n_1 and n_2 be the number of entries whose values are equal to 1 and 2, respectively, (only the entries in the upper triangular region of the matrix are taken into account). The

Table 8
Comparison of QBCA, QBCA(-H) and QBCA(-HS) with respect to the computation time (in seconds).

Data sets	QBCA	QBCA(-H)	QBCA(-HS)
The character trajectories data set	0.0401	0.0424	0.0455
The iris data set	0.00318	0.00323	0.00335
The wine data set	0.00361	0.00376	0.00385

Rand index (RI) [50] is defined as follows:

$$RI = \frac{n_1}{n_1 + n_2} \tag{35}$$

Tables 7 and 8 provide a comparison of the performance of QBCA, QBCA(-H) and QBCA(-HS) in terms of the accuracy and the computation time for the character trajectories data set, the iris data set and the wine data set. From Table 7, it can be seen that QBCA and its variants obtain satisfactory results for the character trajectories data set and the iris data set. The possible reason is that: (i) the CCI stage in QBCA successfully identifies the initial cluster centers which are in the neighborhood of the true cluster centers. (ii) The CCA stage in QBCA correctly assigns most of the data points to their corresponding clusters. QBCA outperforms QBCA(-H) and QBCA(-HS) in terms of the computation time, as shown in Table 8, while QBCA(-H) is still more efficient than QBCA(-HS). This indicates that the shrinking process and the hierarchical structure are two important factors which make QBCA more efficient. However, the results obtained by QBCA in terms of the accuracy are not very satisfactory for the wine data set, as shown in Table 7. The possible reason is that data points belonging to different classes are difficult to be separated from each other in this data set, as shown in Fig. 13(c).

8. Conclusion and future work

In this paper, we investigate the problem of performing efficient clustering on different types of data sets. Although there exist a large number of clustering approaches for different applications, few of these approaches address the issue of improving clustering efficiency. Our major contribution is a newly proposed quantization-based clustering algorithm (QBCA). Specifically, QBCA consists of three stages: (i) the quantization process; (ii) the cluster center initialization stage; and (iii) the cluster center assignment stage. In particular, the cluster center initialization stage is applied to increase the convergence rate, and the cluster center assignment stage is designed to reduce the distance computation cost between the points and the centers of the clusters. QBCA achieves low computation time by combining these three stages. We compare QBCA with different variants of the K-means algorithm based on the kd-tree. In our experiments, we observe that QBCA outperforms these algorithms in terms of efficiency, while maintaining the same cluster qualities. In our future work, we shall investigate how to apply QBCA to perform large-scale clustering for a data mining system.

Acknowledgment

The work described in this paper was partially supported by a grant from the Research Grants Council of Hong Kong Special Administrative Region, China [Project no. CityU 121607], and a grant from the City University of Hong Kong [Project no. 7002374].

Appendix A. Proof of Lemma 1

Lemma 1. *If* $\underline{d}(\mathbf{s}',\mathbf{b}) > \overline{d}^*(\mathbf{s},\mathbf{b})$, *then* $\mathbf{s}'$ *is not the closest seed of a point* $\mathbf{p}$ *which belongs to the histogram bin* $\mathbf{b}$.

Proof. There exists at least one seed $\mathbf{s}$ whose maximum distance to the histogram bin $\mathbf{b}$ is equal to $\overline{d}^*(\mathbf{s},\mathbf{b})$ ($\overline{d}^*(\mathbf{s},\mathbf{b}) = \overline{d}(\mathbf{s},\mathbf{b})$). If the point $\mathbf{p}$ belongs to the histogram bin $\mathbf{b}$, four possible cases will arise:

(i) If the point $\mathbf{p}$ is located at a position which is closest to the seed $\mathbf{s}'$ and furthest from the seed $\mathbf{s}$, the following condition is satisfied: $d(\mathbf{s},\mathbf{p}) = \overline{d}^*(\mathbf{s},\mathbf{b}) < \underline{d}(\mathbf{s}',\mathbf{b}) = d(\mathbf{s}',\mathbf{p})$. As a result, $\mathbf{s}'$ is not the closest seed of the point $\mathbf{p}$.
(ii) If the point $\mathbf{p}$ is located at a position which is closest to the seed $\mathbf{s}'$ but not furthest from the seed $\mathbf{s}$, the following condition is satisfied: $d(\mathbf{s},\mathbf{p}) < \overline{d}^*(\mathbf{s},\mathbf{b}) < \underline{d}(\mathbf{s}',\mathbf{b}) = d(\mathbf{s}',\mathbf{p})$. As a result, $\mathbf{s}'$ is not the closest seed of the point $\mathbf{p}$.
(iii) If the point $\mathbf{p}$ is located at a position which is closest to the seed $\mathbf{s}$ and furthest from the seed $\mathbf{s}'$, the following condition is satisfied: $d(\mathbf{s},\mathbf{p}) < \overline{d}^*(\mathbf{s},\mathbf{b}) < \underline{d}(\mathbf{s}',\mathbf{b}) < d(\mathbf{s}',\mathbf{p})$. As a result, $\mathbf{s}'$ is not the closest seed of the point $\mathbf{p}$.
(iv) If the point $\mathbf{p}$ is located at a position which is closest to the seed $\mathbf{s}$ but not furthest from the seed $\mathbf{s}'$, the following condition is satisfied: $d(\mathbf{s},\mathbf{p}) < \overline{d}^*(\mathbf{s},\mathbf{b}) < \underline{d}(\mathbf{s}',\mathbf{b}) \le d(\mathbf{s}',\mathbf{p})$. As a result, $\mathbf{s}'$ is not the closest seed of the point $\mathbf{p}$.

Lemma 1 is thus proved. □

Appendix B. Proof of Lemma 2

Lemma 2. *For an arbitrary point* $\mathbf{p}$ *which belongs to the histogram bin* $\mathbf{b}$, *the closest seed* $\mathbf{s}$ *to the point* $\mathbf{p}$ *is in the seed candidate list* $\mathbf{b}$.*list of the histogram bin* $\mathbf{b}$.

Proof. According to Lemma 1, if the seed $\mathbf{s}'$ satisfies $\underline{d}(\mathbf{s}',\mathbf{b}) > \overline{d}^*(\mathbf{s},\mathbf{b})$, it is not the closest seed of the point $\mathbf{p}$ which belongs to the histogram bin $\mathbf{b}$. After removing all the seeds which satisfy Lemma 1, the closest seed of the point $\mathbf{p}$ will be among one of those in the remaining seeds. Because the remaining seeds satisfy $\underline{d}(\mathbf{s},\mathbf{b}) \le \overline{d}^*(\mathbf{s},\mathbf{b})$ and the seed candidate list $\mathbf{b}.list$ of the histogram bin $\mathbf{b}$ only stores the seed candidates which satisfy $\underline{d}(\mathbf{s},\mathbf{b}) \le \overline{d}^*(\mathbf{s},\mathbf{b})$ and $\overline{d}^*(\mathbf{s},\mathbf{b}) = \min_{h\in\{1,\ldots,k\}} \overline{d}(\mathbf{s}_h,\mathbf{b})$, $\mathbf{b}.list$ will contain all the remaining seeds. This implies that the seed which is closest to the point $\mathbf{p}$ is in $\mathbf{b}.list$. As a result, Lemma 2 is correct. □

Appendix C. Proof of Lemma 3

Lemma 3. *If* $\mathbf{b} \subseteq \mathbf{b}^{parent}$, *the seed candidates of the child histogram bin* $\mathbf{b}$ *is a subset of the seed candidates of the parent histogram bin* $\mathbf{b}^{parent}$:

$$\mathbf{b}.list \subseteq \mathbf{b}^{parent}.list \tag{36}$$

where $\mathbf{b}.list$ and $\mathbf{b}^{parent}.list$ *denote the seed candidate list of the child histogram bin and the parent histogram bin, respectively.*

Proof. According to the definition of the seed candidate list, (i) all the seed candidates in the seed candidate list $\mathbf{b}.list$ of the child histogram bin $\mathbf{b}$ satisfy $\underline{b}(\mathbf{s},\mathbf{b}) \le \overline{d}^*(\mathbf{s},\mathbf{b})$ and $\overline{d}^*(\mathbf{s},\mathbf{b}) = \min_{h\in\{1,\ldots,k\}} \overline{d}(\mathbf{s}_h,\mathbf{b})$; (ii) all the seed candidates in the seed candidate list $\mathbf{b}^{parent}.list$ of the parent histogram bin $\mathbf{b}^{parent}$ satisfy $\underline{d}(\mathbf{s},\mathbf{b}^{parent}) \le \overline{d}^*(\mathbf{s},\mathbf{b}^{parent})$ and $\overline{d}^*(\mathbf{s},\mathbf{b}^{parent}) = \min_{h\in\{1,\ldots,k\}} \overline{d}(\mathbf{s}_h,\mathbf{b}^{parent})$. Since $\mathbf{b} \subseteq \mathbf{b}^{parent}$, $\overline{d}^*(\mathbf{s},\mathbf{b}) \le \overline{d}^*(\mathbf{s},\mathbf{b}^{parent})$ and $\underline{d}(\mathbf{s},\mathbf{b}) \ge \underline{d}(\mathbf{s},\mathbf{b}^{parent})$, the following relationship is deduced for all the seed candidates

in $\mathbf{b}.list$: $\underline{d}(\mathbf{s},\mathbf{b}^{parent}) \le \underline{d}(\mathbf{s},\mathbf{b}) \le \overline{d}^{*}(\mathbf{s},\mathbf{b}) \le \overline{d}^{*}(\mathbf{s},\mathbf{b}^{parent})$. As a result, $\mathbf{b}.list \subseteq \mathbf{b}^{parent}.list$, and Lemma 3 is correct. □

Appendix D. Proof of Corollary 1

Corollary 1 (*Hierarchical relationship of the candidate list*). *Let* $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_E\}$ *be a set of histogram bins and* $\{\mathbf{b}_1.list, \mathbf{b}_2.list, \ldots, \mathbf{b}_E.list\}$ *be the corresponding seed candidate lists of the histogram bins* (*where E is the total number of histogram bins in the set*). *If the histogram bins satisfy the hierarchical relationship* $\mathbf{b}_1 \supseteq \mathbf{b}_2 \supseteq \cdots \supseteq \mathbf{b}_E$, *the seed candidate lists of the histogram bins satisfy the hierarchical relationship* $\mathbf{b}_1.list \supseteq \mathbf{b}_2.list \supseteq \cdots \supseteq \mathbf{b}_E.list$.

Proof. Since $\mathbf{b}_E \subseteq \mathbf{b}_{E-1}$, …, $\mathbf{b}_3 \subseteq \mathbf{b}_2$ and $\mathbf{b}_2 \subseteq \mathbf{b}_1$, the following relationship is deduced according to Lemma 3: $\mathbf{b}_E.list \subseteq \mathbf{b}_{E-1}.list$, …, $\mathbf{b}_3.list \subseteq \mathbf{b}_2.list$ and $\mathbf{b}_2.list \subseteq \mathbf{b}_1.list$. As a result, Corollary 1 is correct. □

Appendix E. Proof of Lemma 4

Lemma 4. *The hierarchical relation of the candidate list is still satisfied after applying the shrinking process to the hierarchical structure*.

Proof. We assume there exists two bins $\mathbf{b}^{e+1}$ and $\mathbf{b}^{e}$ such that they satisfy the condition $\mathbf{b}^{e+1} \subseteq \mathbf{b}^{e}$. According to the property of the shrinking process, $\mathbf{b}'^{e+1} \subseteq \mathbf{b}'^{e}$ (where $\mathbf{b}'^{e+1}$ and $\mathbf{b}'^{e}$ denote the bins after shrinking). According to Lemma 3, $\mathbf{b}'^{e+1}.list \subseteq \mathbf{b}'^{e}.list$. Let $\{\mathbf{b}_1, \mathbf{b}_2 \ldots \mathbf{b}_E\}$ be a set of bins in the hierarchical structure and $\{\mathbf{b}_1.list, \mathbf{b}_2.list \ldots \mathbf{b}_E.list\}$ be the corresponding seed candidate lists of the bins. If the bins before shrinking satisfy the relation $\mathbf{b}_1 \supseteq \mathbf{b}_2 \supseteq \cdots \supseteq \mathbf{b}_E$, the bins after shrinking will still satisfy the relation $\mathbf{b}'_1 \supseteq \mathbf{b}'_2 \supseteq \cdots \supseteq \mathbf{b}'_E$. As a result, the seed candidate lists of the bins after shrinking satisfy the hierarchical relationship $\mathbf{b}'_1.list \supseteq \mathbf{b}'_2.list \supseteq \cdots \supseteq \mathbf{b}'_E.list$ according to Corollary 1. As a result, Lemma 4 is correct. □

References

[1] S. Guha, R. Rastogi, K. Shim, CURE: an efficient clustering algorithm for large data sets, in: Proceedings of the ACM SIGMOD Conference, 1998, pp. 73–84.
[2] A. Hinneburg, D.A. Keim, An efficient approach to clustering in large multimedia databases with noise, in: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, 1998, pp. 58–65.
[3] S. Guha, R. Rastogi, K. Shim, ROCK: a robust clustering algorithm for categorical attributes, in: Proceedings of the IEEE Conference on Data Engineering, 1999, pp. 345–366.
[4] C.Y. Chen, S.C. Hwang, Y.-J. Oyang, An incremental hierarchical data clustering algorithm based on gravity theory gravity theory, in: Advances in Knowledge Discovery and Data Mining: Sixth Pacific-Asia Conference, PAKDD 2002, Taipei, Taiwan, May 6–8, 2002, pp. 237–250.
[5] T. Zhang, R. Ramakrishnan, M. Linvy, BIRCH: an efficient data clustering method for very large data sets, Data Mining and Knowledge Discovery 1 (2) (1997) 141–182.
[6] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases, in: Proceedings of the 1996 International Conference on Knowledge Discovery and Data Mining (KDD'96), Portland, Oregon, August 1996, pp. 226–231.
[7] M. Ankerst, M. Breunig, H.-P. Kriegel, J. Sander, Optics: ordering points to identify the clustering structure, in: Proceedings of the 1999 ACM-SIGMOD International Conference on Management of Data, June 1999, pp. 49–60.
[8] W. Wang, J. Yang, R. Muntz, STING: a statistical information grid approach to spatial data mining, in: Proceedings of the 23rd VLDB Conference, Athens, Greece, 1997, pp. 186–195.
[9] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data for data mining applications, in: Proceedings of the ACM-SIGMOD International Conference on Management of Data, June 1998, pp. 94–105.
[10] T. Kanungo, D.M. Mount, N.S. Netanyahu, An efficient k-means clustering algorithm: analysis and implementation, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7) (2002) 881–892.
[11] M.C. Su, C.H. Chou, A modified version of the K-means algorithm with a distance based on cluster symmetry, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (6) (2001) 674–680.
[12] J.Z. Huang, M.K. Ng, H.Q. Rong, Z.C. Li, Automated variable weighting in k-means type clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (5) (2005) 657–668.
[13] D. Charalampidis, A modified K-means algorithm for circular invariant clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (12) (2005) 1856–1865.
[14] F. Camastra, A. Verri, A novel kernel method for clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (5) (2005) 801–805.
[15] P.S. Bradley, U. Fayyad, Refining initial points for K-means clustering, in: Proceedings of the 15th International Conference on Machine Learning, 1998, pp. 91–99.
[16] S.P. Lloyd, Least squares quantization in PCM, IEEE Transactions on Information Theory 28 (1982) 129–137.
[17] S. Kolliopoulos, S. Rao, A nearly linear-time approximation scheme for the Euclidean k-median problem, in: Proceedings of the Seventh Annual European Symposium on Algorithms, July 1999, pp. 362–371.
[18] J. MacQueen, Some methods for classification and analysis of multivariate observations in: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, 1967, pp. 281–296.
[19] S.P. Lloyd, An algorithm for vector quantizer design, IEEE Transactions on Communications 28 (1) (1982) 84–95.
[20] Z. Huang, Extensions to the k-means algorithms for clustering large data sets with categorical values, Data Mining and Knowledge Discovery 2 (3) (1998) 283–304.
[21] J. Yin, Z.F. Tan, J.T. Ren, Y.Q. Chen, An efficient clustering algorithm for mixed type attributes in large data set, in: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, vol. 3, August 2005, pp. 1611–1614.
[22] T. Kanungo, D.M. Mount, N.S. Netanyahu, C. Piatko, R. Silverman, A.Y. Wu, Computing nearest neighbors for moving points and applications to clustering, in: Proceedings of the 10th Annual ACM–SIAM Symposium on Discrete Algorithms, January 1999, pp. 931–932.
[23] K. Alsabti, S. Ranka, V. Singh, An efficient k-means clustering algorithm, in: Proceedings of the First Workshop High Performance Data Mining, March 1998, pp. 881–892.
[24] D. Pelleg, A. Moore, Accelerating exact k-means algorithms with geometric reasoning, in: Proceedings of the ACM SIGKDD International Conference Knowledge Discovery and Data Mining, August 1999, pp. 277–281.
[25] D. Pelleg, A. Moore, X-means: extending k-means with efficient estimation of the number of clusters, in: Proceedings of the 17th International Conference Machine Learning, July 2000, pp. 727–734.
[26] L. Bottou, Y. Bengio, Convergence properties of the k-means algorithms, in: G. Tesauro, D. Touretzky (Eds.), Advances in Neural Information Processing Systems, vol. 7, MIT Press, Cambridge, Massachusetts, 1995, pp. 585–592.
[27] D. Pollard, A central limit theorem for k-means clustering, Annals of Probability 10 (1982) 919–926.
[28] S.Z. Selim, M.A. Ismail, K-means-type algorithms: a generalized convergence theorem and characterization of local optimality, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (1984) 81–87.
[29] J. Matousek, On approximate geometric k-clustering, Discrete and Computational Geometry 24 (2000) 61–84.
[30] O.L. Mangasarian, Mathematical programming in data mining, Data Mining and Knowledge Discovery 1 (1997) 183–201.
[31] F. Pernkopf, D. Bouchaffra, Genetic-based EM algorithm for learning Gaussian mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (8) (2005) 1344–1348.
[32] S. Lazebnik, C. Schmid, J. Ponce, A maximum entropy framework for part-based texture and object recognition, in: Proceedings of the IEEE International Conference on Computer Vision, vol. 1, October 2005, Beijing, China, pp. 832–838.
[33] S. Wang, J.M. Siskind, Image segmentation with ratio cut, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (6) (2003) 675–690.
[34] J.Q. Liu, Y.H. Yang, Multiresolution color image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (7) (1994) 689–700.
[35] A. Neumann, Graphical Gaussian shape models and their application to image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (3) (2003) 316–329.
[36] L. Grady, Random walks for image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (11) (2006) 1768–1783.
[37] J.C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, Journal of Cybernetics 3 (3) (1973) 32–57.
[38] J. Yu, General C-means clustering model, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (8) (2005) 1197–1211.
[39] Y. Cheng, Mean shift, mode seeking, and clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (8) (1995) 790–799.
[40] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (5) (2002) 603–619.

[41] M. Laszlo, S. Mukherjee, A genetic algorithm using hyper-quadtrees for low-dimensional K-means clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (4) (2006) 533–538.
[42] M. Laszlo, S. Mukherjee, A genetic algorithm that exchanges neighboring centers for k-means clustering, Pattern Recognition Letters 28 (16) (2007) 2359–2366.
[43] S. Bandyopadhyay, S. Saha, GAPS: a clustering method using a new point symmetry-based distance measure, Pattern Recognition 40 (12) (2007) 3430–3451.
[44] K.L. Chung, J.S. Lin, Faster and more robust point symmetry-based K-means algorithm, Pattern Recognition 40 (2) (2007) 410–422.
[45] J.Z.C. Lai, Y.C. Liaw, Improvement of the k-means clustering filtering algorithm, Pattern Recognition 41 (12) (2008) 3677–3681.
[46] J.Z.C. Lai, T.J. Huang, Y.C. Liaw, A fast k-means clustering algorithm using cluster center displacement, Pattern Recognition 42 (11) (2009) 2551–2556.
[47] D.X. Chang, X.D. Zhang, C.W. Zheng, A genetic algorithm with gene rearrangement for K-means clustering, Pattern Recognition 42 (7) (2009) 1210–1222.
[48] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Computing Surveys 31 (3) (1999) 264–323.
[49] A. Asuncion, D.J. Newman, UCI Machine Learning Repository 〈http://www.ics.uci.edu/∼ mlearn/MLRepository.html〉, University of California, School of Information and Computer Science, Irvine, CA, 2007.
[50] W.M. Rand, Objective criteria for the evaluation of clustering methods, Journal of the American Statistical Association 66 (1971) 846–850.

About the Author—ZHIWEN YU received BSc and Master degrees from Sun Yat-Sen University in China, and PhD degree from City University of Hong Kong. He is currently a Research Fellow at the Department of Computer Science, City University of Hong Kong. His research interests include multimedia, image processing, machine learning and data mining.

About the Author—HAU-SAN WONG is currently an Associate Professor in the Department of Computer Science, City University of Hong Kong. He received BSc and MPhil degrees in Electronic Engineering from the Chinese University of Hong Kong, and PhD degree in Electrical and Information Engineering from the University of Sydney. He has also held research positions in the University of Sydney and Hong Kong Baptist University. His research interests include multimedia signal processing, neural networks and evolutionary computation. He is the co-author of the book *Adaptive Image Processing*: *A Computational Intelligence Perspective*, which is a joint publication of CRC Press and SPIE Press, and was an organizing committee member of the 2000 IEEE Pacific-Rim Conference on Multimedia and 2000 IEEE Workshop on Neural Networks for Signal Processing, which were both held in Sydney, Australia. He has also co-organized a number of conference special sessions, including the special session on "Image Content Extraction and Description for Multimedia" in 2000 IEEE International Conference on Image Processing, Vancouver, Canada, and "Machine Learning Techniques for Visual Information Retrieval" in 2003 International Conference on Visual Information Retrieval, Miami, Florida.