Clusterability Detection and Cluster Initialization

Scott Epter [*], Mukkai Krishnamoorthy, Mohammed J. Zaki [†]
Computer Science Department
Rensselaer Polytechnic Institute, Troy, NY 12180
septer@us.ibm.com, {moorthy,zaki}@cs.rpi.edu

Abstract

The need for a preliminary assessment of the clustering tendency or clusterability of massive data sets is known. A good clusterability detection method should serve to influence a decision as to whether to cluster at all, as well as provide useful seed input to a chosen clustering algorithm. We present a framework for the definition of the clusterability of a data set from a distance-based perspective. We discuss a graph-based system for detecting clusterability and generating seed information including an estimate of the value of k – the number of clusters in the data set, an input parameter to many distance-based clustering methods. The output of our method is tunable to accommodate a wide variety of clustering methods.

We have conducted a number of experiments using our methodology with stock market data and with the well-known BIRCH data sets, in two as well as higher dimensions. Based on our experiments and results we find that our methodology can serve as the basis for much future work in this area. We report our results and discuss promising future directions.

1 Introduction

Much work has been done recently in the area of unsupervised learning or *data clustering*. Data clustering algorithms find the natural groupings or *clusters* in a set of data. They are distinguished from the supervised learning or *classification* algorithms, which attempt to assign the elements of a set of data to a group of predefined classes.

One method of detecting clusters is to view the elements of the data set as points in a metric space in which some distance metric is defined as in [18]. A cluster can then be informally defined as a set of points that are mutually 'close' together, where two points are considered close if the metric difference between them is small. Clustering quality is determined by the minimization of *intra*-cluster distance and/or the maximization of *inter*-cluster distance. Intuitively, a good clustering will have a high degree of distance among points in different clusters (*separation*) and a low degree of distance among points in the same cluster (*distortion*). Distance-based algorithms perform quite well when working with data sets that contain very distinct, and in general spherically-shaped groups of points. Most distance-based algorithms are variants of the classic k-means algorithm, in which part of the input to the algorithm is a value k that signifies the desired number of clusters. Each cluster is represented by a single centroid. As each point is scanned it is added to the cluster of the centroid to which it is closest. Distance-based approaches have been the traditional approach to analyzing numeric data.

Recently, several works have been published that explore cases in which distance-based methods are less effective, in particular when data clusters naturally are of irregular often non-convex shapes and are close together in the space. Mining patterns in such *spatial* data is more akin to detecting clusters as they would be seen by humans.

In this regard there have emerged two clearly-defined, distinct approaches to clustering data: distance-based and spatial. To illustrate this idea, consider the dataset shown in Figure 1. When viewed as a set of *spatial* data, there are two distinctly discernible clusters of points. The regions in between the two clusters

[*] Currently with IBM Power Parallel Group in Poughkeepsie, NY

[†] Supported in part by NSF CAREER Award IIS-0092978, and NSF Next Generation Software Program grant EIA-0103708.

are sufficiently sparse as to separate them as objects but not much more than that. In this context, points X and W belong in one cluster, while Y and Z belong in the other.

It would not be desirable for a distance-based method to cluster this data in a similar manner. Such an approach would be more likely to group X with Y and W with Z. Depending on the desired end-result knowledge, it may very well make sense to consider both objects together as comprising a single large cluster since the extent of overlap between the two is much more pronounced than the small degree of separation. Suppose for example that this data represented some sort of demographic customer profile. The most useful information attainable from this dataset may be that there is no clearly discernible pattern whatsoever.

In this paper we discuss perspectives on distance-based clustering, providing means for evaluating the distance-based clusterability of an individual data set. Our work is a fresh approach to what is referred to in [16] as *clustering tendency*. Given a dataset, we discern whether a distance-based approach is likely to yield fruitful results. In so doing, we generate seed centroids likely to improve the accuracy and order-stability of distance-based clustering algorithms. Our method is tunable so that the type of output can be geared toward specific algorithms and is shown to be efficient in high dimensions. Experimental results indicate a high-degree of accuracy as well as sub-linear slowdown, even in massive (1000) dimensions.

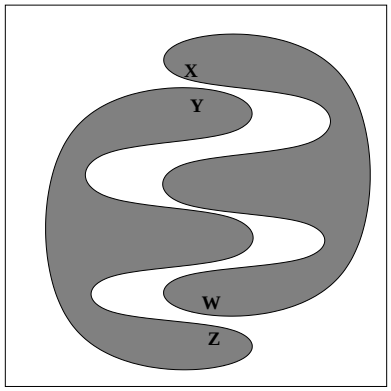

Figure 1: A subjectively-clusterable dataset.

There are two main parts of our work. First, we determine the clusterability of a data set by examining the histogram of point-to-point distances. We then use the histogram to determine *seed* information for any k-means algorithm, in particular a value for k as well as an estimate of the k centroid points. For multiple-centroid methods, e.g. [12] the sensitivity of our detection method can be tuned to generate an initial set of multiple centroids.

We determine the seed information with various graph-theoretic formulations. In the past, graph theory has been used in identifying clusters [5, 13, 16, 19]. The approaches that have been taken so far treat each point as a node in a graph and define adjacency relationship among nodes based on some distance metric. Two common ideas treat clusters as approximate cliques or use algorithms to partition nodes of graphs (such that each partition is treated as an approximate cluster).

We propose a different definition of adjacency relationship. We determine adjacency between two nodes not directly but indirectly by obtaining a histogram of all pair edge lengths and then, using this histogram, we determine a threshold for the intra-cluster distance. By considering the intra-cluster nodes (corresponding to those points) and their corresponding edges, we get a subgraph. By analyzing the connected components [6], we get a starting point for the clusters. The thrust of our approach is to identify proper starting points for any of the clustering methods. In this fashion, we find the application of graph theory to be useful and to provide insight.

The rest of this paper is as follows: in Section 2 we discuss related work. In Section 3 we define the basis of our framework. In Section 3.2 we weave the definitions presented in Section 3 into the theory behind our method and then in Section 4 we discuss the application of the theory. In Section 5 we explain our experiments and results. Finally in Section 6 we offer concluding remarks and prospects for future work.

2 Related Work

The detection of groups in data has been studied for decades in several domains. [7, 16, 17] provide excellent overviews of standard issues in data clustering, as well as classical approaches to handling them. The recent book by Han and Kamber [14] also provides a much needed reference on the latest results on clustering from the data mining community.

Recent attention in the context of data mining has focussed largely on the challenges imposed by large datasets and high dimensionality. Several works [4, 12, 19, 21] are based on traditional distance-based clustering, and in particular the k-means algorithm, in which the number of desired clusters k is given as part of the input. As mentioned above, such methods require the existence of a distance *metric* between points in the space, where the most common metrics are Euclidean and Manhattan distance. In general, k-means-type algorithms make extensive use of a *centroid* for each cluster, a single point that represents the entire cluster.

A common means of determining the dimensions of a centroid is to average the values of the dimensions of each of the points in the cluster.

CLARANS [19] views the solution space as a large graph, in which each node of the graph represents the current set of centroids. The graph is traversed and a node is visited if it will improve upon the objective distance function. BIRCH [21] introduces the novel approach of minimizing the amount of information stored with each cluster and demonstrates substantial improvements in both runtime and accuracy over [19]. A common characteristic of these two systems is that they are dependent on the fact that the clusters are spherically-shaped. CURE [12] introduces a *multiple*-centroid approach and is particularly effective at discovering elliptical-shaped clusters. [4] describes a k-means type algorithm that improves on the stored information-reduction approach first described in [21] and can determine clusters in a single-scan of the data, a particularly desirable characteristic in the presence of very large data sets.

Works have also emerged that are specifically geared toward *spatial* data[8, 20], the clusters of which are generally irregularly-shaped and can be close together, compromising the effectiveness of distance-based methods. WaveCluster[8, 20] use notions of point density to detect cluster boundaries and thus the clusters themselves. Another class of clustering algorithms perform subspace or projective clustering [1, 2], i.e., instead of using all the dimensions they find clusters in dense subspaces. This is particularly effective in high dimensional spaces where it becomes hard to distinguish between the nearest and farthest neighbors of a data point [15]. By looking at important subspaces meaningful clusters can be found.

Our work focuses on open problems in the area of distance-based clustering. A common theme among the current algorithms is the need to *seed* the system. At the very least, the value of k is required, but in general, both performance and accuracy improvements can be realized if seed information is provided[3, 4, 9]. Distance-based algorithms, especially those that make only a few database scans, are particularly volatile in their earliest iterations, before a stable state is reached. Our work provides an effective solution to this problem as a means of working in conjunction with existing algorithms. Our assumption is that an initial value of k as well as an initial stable set of cluster centroids will greatly increase the stability of such algorithms.

A different approach to a similar problem is taken in [3, 9]. A statistical method is presented that seeds clustering algorithms by discovering clusters in several small sub-samples of the data. To the best of our knowledge this is the only available work that deals with initial seeding of k-means algorithms. Our work differs from the work in [3, 9] in several ways:

- We provide an means for analyzing clustering tendency, or *clusterability* in a data set. This information is used before any attempt to detect the clusters in the database.
- Our approach to seed generation is based on this clusterability information. We use graph-theoretic constructs to generate approximations of cluster centroids based on 'pretending' that the data set is of high clusterability.
- Part of our *output* is a value for k, the number of expected clusters. Part of the *input* to the system described in [3, 9] is a value for k; thus it is expected that our system could work in conjunction with that proposed in [3, 9], rather than in place of it.

Having established the basis of the problem space and the currently available solutions we now turn our attention to defining our framework for clusterability detection, beginning with a set of definitions.

3 Definitions and Framework

In this section we present the definitions that are the basis of our framework. We begin by clearly defining the components of a clustering and then build upon these definitions to establish notions of clustering tendency.

A data set $\mathcal{D}$ is a unique, fixed set of n-dimensional points in a metric space. The number of points in a data set $\mathcal{D}$ is denoted $|\mathcal{D}|$. A *cluster* C of a data set $\mathcal{D}$ is a subset of $\mathcal{D}$. A *clustering* $\mathcal{C}$ of a data set $\mathcal{D}$ is a set of distinct clusters $\{C_1, C_2, \ldots, C_k\}$ such that $\bigcup_{i=1}^{k} C_i = \mathcal{D}$. A *trivial clustering* $\mathcal{C}$ with k clusters of a data set $\mathcal{D}$ is one in which either $k = 1$ or $k = |\mathcal{D}|$, i.e., the case where all points are in one cluster ($k = 1$), and the case where each point is in its own cluster ($k = |\mathcal{D}|$). A clustering which is not trivial is said to be non-trivial. If $k = O(n)$ we call the clustering *near-trivial*. A *useful clustering* $\mathcal{C}$ with k clusters of a data set $\mathcal{D}$ is one which is nontrivial and $k = O(1)$. Note that if $k = O(n)$ then we have that $k = O(1)$ ensure that a useful clustering avoids being near-trivial.

Definition 3.1 [Properly Clusterable Data Set] A properly clusterable data set is a data set $\mathcal{D}$ for which there exists a useful clustering $\mathcal{C}$ composed of clusters $C_1, C_2, \ldots, C_k$ such that, for each point $p_i \in C_i$, p_i is closer to every point in $\mathcal{C}_i$ then to any point not in $\mathcal{C}_i$.

The clustering $\mathcal{C}$ is known as a *proper clustering* of the dataset $\mathcal{D}$. Definition 3.1 ensures that, for a data set to be considered properly clusterable, there must exist a non-trivial clustering for which each *point* clearly belongs to the cluster to which it is assigned and there is no possibility that it would be assigned to any other.

Definition 3.2 [Edge] An edge is a four-tuple: $\{u, v, \Delta_{u,v}, \omega_{\mathcal{C}}\}$, where $\mathcal{C}$ is a clustering, u and v are points in a data set, $\Delta_{u,v}$ is the distance[1] between u and v and:

$$\omega_{\mathcal{C}} = \begin{cases} \text{internal} & \text{if } u \text{ and } v \text{ belong to the same cluster in } \mathcal{C} \\ \text{external} & \text{otherwise} \end{cases}$$

Since each edge is uniquely determined by the points which it represents, we will use the notation $E_{u,v}$ to denote the edge between points u and v. Also, unless otherwise specified, the weight of $E_{u,v}$, $\Delta_{u,v}$ is fixed.

Edges for which $\omega_{\mathcal{C}} =$ external are referred to as *external* with respect to $\mathcal{C}$, and similarly for *internal edges*. For a data set in which no points have been assigned to any clusters ($\mathcal{C} = \phi$), all edges are by Definition 3.2 external.

Definition 3.3 [Perfectly Clusterable Data Set] A perfectly clusterable data set is a properly clusterable data set for which there exists a "useful clustering" $\mathcal{C}$, such that for all edges $\{u, v, \Delta_{u,v}, \text{internal}_{\mathcal{C}}\}$ and all edges $\{w, x, \Delta_{w,x}, \text{external}_{\mathcal{C}}\}$, $\Delta_{u,v} < \Delta_{w,x}$.

The clustering $\mathcal{C}$ is known as a *perfect clustering* of $\mathcal{D}$. Definition 3.3 ensures of a perfectly clusterable data set that there exists a clustering for which each *edge* belongs unambiguously to either the set of internal or external edges. A properly clusterable dataset which is not perfectly clusterable would have one or more clusters with large distance between points in the same cluster. Given a perfectly clusterable data set, any k-means based algorithm should be able to cluster the points into distinct clusters in one pass over the data set.

A *cut-off point* for a Perfectly clusterable data set is a distance c such that for all edges $\{u, v, \Delta_{u,v}, \text{internal}_{\mathcal{C}}\}$ and all edges $\{w, x, \Delta_{w,x}, \text{external}_{\mathcal{C}}\}$, $\Delta_{u,v} < \ c \ < \Delta_{w,x}$. In other words a good cut-off point results in well separated clusters. Finding such a cut-off point is a nontrivial task. In our algorithms, we suggest way for calculating c (in Section 5.2.2).

3.1 Examples

Here we present illustrative examples of the types of data sets and clusterings defined above.

Figure 2A) shows a data set that is not properly clusterable. Clustering this data set is somewhat subjective, and different interpretations will result in different clusterings. Consider points a, b, and c, with c and a both exactly one unit of distance apart from b. Do a and b belong together, or b and c? An argument could be made for either of these options, as well as the case that neither or all of them end up together.

Figure 2B) shows a properly clusterable data set. There is little argument as to which points belong together in which clusters. However, there is no non-trivial clustering which exhibits a clean schism between the internal and external edge *lengths*. In other words, there is no clearly-defined threshold as to what constitutes an internal edge and what constitutes an external edge.

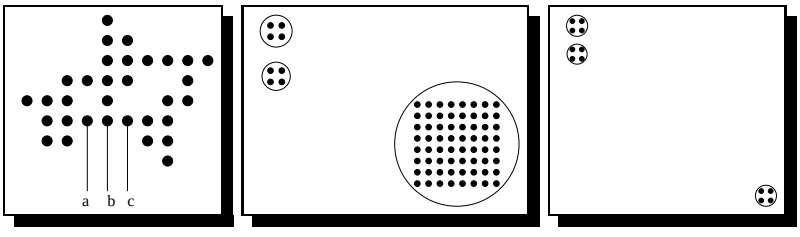

Figure 2: A) Non-properly-clusterable dataset, B) Properly-clusterable data set, C) Perfectly-clusterable dataset.

Figure 2C) shows a perfectly clusterable data set. Given any edge e between two points, there is no question

[1] distance here can denote any metric satisfying the definition in [18]. In general we will make use of either Euclidean or Manhattan distance.

as to whether e is internal or external. Intuitively this type of data set represents the absolute best-case of *clusterability*.

3.2 Framework

In this section, we explain the theory behind our work. For a perfectly clusterable data set $\mathcal{D}$, the clear distinction between the set of internal edges and the set of external edges can be exploited as follows:

Let us represent $\mathcal{D}$ as a completely-connected, undirected, weighted graph $\mathcal{G}$ where the nodes of $\mathcal{G}$ represent the points in the data set and the edges of $\mathcal{G}$ represent the edges of $\mathcal{D}$ as defined in Section 3[2].

Consider the perfect clustering $\mathcal{C}$ of $\mathcal{D}$. Since the longest internal edge of $\mathcal{D}$ is shorter than the shortest external edge, there exists a cut-off value c, such that any edge of length $<= c$ is internal, and any edge of length $> c$ is external. Determining the clusters in this respect is as simple as casting $\mathcal{G}$ as a completely-connected *signed* graph[3] $\mathcal{G}^{'}$ with node set equivalent to that of $\mathcal{G}$. The edge set $E^{'}$ of $\mathcal{G}^{'}$ is constructed from the edge set E of $\mathcal{G}$ as follows, for all $(E_{u,v} \in E)$,

$$E^{'}_{u,v} = \begin{cases} + & \text{if } \Delta_{u,v} \leq c \\ - & \text{otherwise} \end{cases}$$

The problem of determining the clusters in $\mathcal{D}$ is then equivalent to the clustering problem described in [5], i.e., finding the clusters in $\mathcal{D}$ reduces to the problem of finding sets of nodes in $\mathcal{G}^{'}$ such that for any two nodes u, v in the same cluster, $E^{'}_{u,v} = +$ and for all pairs of nodes w, x in different clusters, $E^{'}_{w,x} = -$.

The main point is how to choose the cut-off distance c? Consider the data distribution of $\Delta_{u,v}$ for all $u, v \in \mathcal{D}$, or more formally the distribution of the probability that distance (or equivalently, the edge lengths) between pairs of points u and v is less than c, i.e. $P[\Delta_{u,v} \leq c]$. The idea behind our method is to determine the clusterability information (or the probability density) for a data set using a histogram of the edge lengths, and then to use this information to find an actual or approximate value for the cutoff c, most likely the bucket where the first clearly defined spike reaches its minimum. A similar approach was taken in [15] to find the quality of a subspace projection for nearest neighbor queries in high dimensions. For example, in the histogram of Figure 5, an exact cutoff exists at bucket 4, the minimum of the spike from buckets $0 - 4$ and thus this would be the chosen value for c. We then generate the graph $\mathcal{G}$ and do a depth-first-traversal of $\mathcal{G}$, restricting adjacency to those points u, v, such that $\Delta_{u,v} \leq c$. The points of each connected path generated in this traversal are averaged and the result is output as an approximate centroid in the set. Note that this restricted traversal is equivalent to a depth-first traversal on the graph $\mathcal{G}^{'}$.

The complete algorithm for clusterability detection and seed-generation is presented in Figure 4.2. We now turn our attention to a discussion of clusterability detection. Jain and Dubes describe three classifications of clustering tendency in data: *uniform* in which points exhibit "mutual repulsion", *random* in which points exhibit no clear-cut tendency, and *clusterable* in which points exhibit "mutual cohesion"[16].

Perfectly clusterable data sets as we define them are a concrete, finer-grained distinction of the third type of data set in Jain and Dubes' classification. The three classes of clusterability defined above are still characterized by subjective interpretation, whereas our notion of perfectly clusterable is mathematically precise.

Our hypothesis, confirmed through experimentation, is that the three high-level classes of clustering tendency are revealed by the structure of the edge-length frequency histogram for the data set in question. Our claim is that the existence and characterization of extreme points in the histogram reveals the non-uniformity and mutual attraction of points in the space. We supply supporting examples in the next section.

4 Methodology and Examples

The purpose of this section is to explain the bridge between the theory described in Section 3.2 and our methods for determining clusterability and generating seed information. We begin with an explanation of the various classifications of clusterability, continue with a high-level description of our application, and conclude with a description of the complete algorithm.

[2]Note that unless otherwise noted the weight of $E_{u,v}$, the quantity $\Delta_{u,v}$, the distance between u and v and the length of the edge between u and v are for all intents and purposes equivalent and the terms are thus used interchangeably.

[3]Recall that a signed graph is one in which each edge is either positive or negative.

4.1 Motivating Examples

The basis of our clusterability determination comes from an analysis of the all-pairs edge length frequency histogram of a data set. Exactly how each histogram is generated is described in Section 4.2.1. For the moment we turn our attention to examples of data sets and their histograms, beginning with a short description of histogram structure. For each example, notions of how the histogram reveals the clusterability of the underlying data set are also explained.

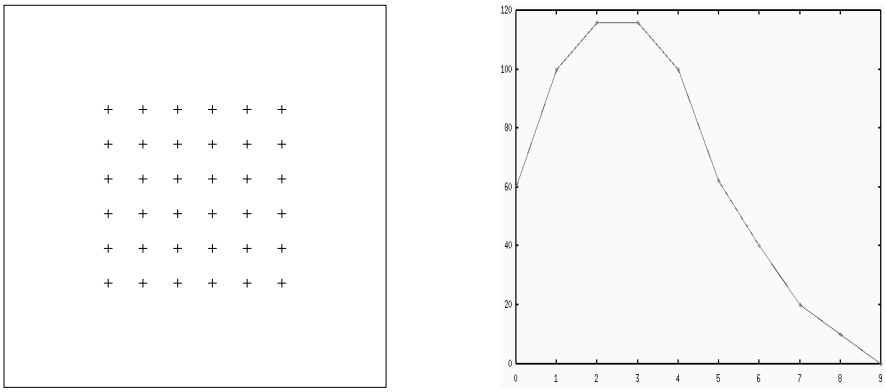

Figure 3: Uniform data set, Edge frequency histogram.

Each bucket in the histogram represents the frequency of occurrence of edge lengths over a specific interval, where each length is normalized over the interval between the minimum E_{min} and maximum E_{max} lengths in the set. For the sake of clarity in the examples, unless otherwise noted it is assumed that distance refers to Manhattan distance.

Suppose then that there are b buckets for a given histogram. The bucket index values range from $[0 \ldots b-1]$. Each edge length is normalized to a value on the interval $[0 \ldots 1]$ and an index is determined by multiplying the normalized value by b in order to select a bucket in the range $[0 \ldots b-1]$. A counter is maintained for each bucket. The counter of bucket i is incremented each time a normalized edge length 'lands' in bucket i. In the histogram plots, each plotted point corresponds to a single bucket, where the x value of the plotted point is the bucket index and the y value is the final total of the counter for bucket x.

Figure 3 shows a perfectly uniform, six-by-six set of data points, as well as the corresponding edge-frequency histogram. Each data point in the set is exactly one unit away from its nearest neighbor, two units away from its nearest neighbor's neighbors[4], etc. The important element to notice about the plot is that the entire range of buckets is represented; there are no gaps. The conspicuous absence of irregularity in the histogram curve mirrors the complete absence of irregularity of the edge lengths in the data set.

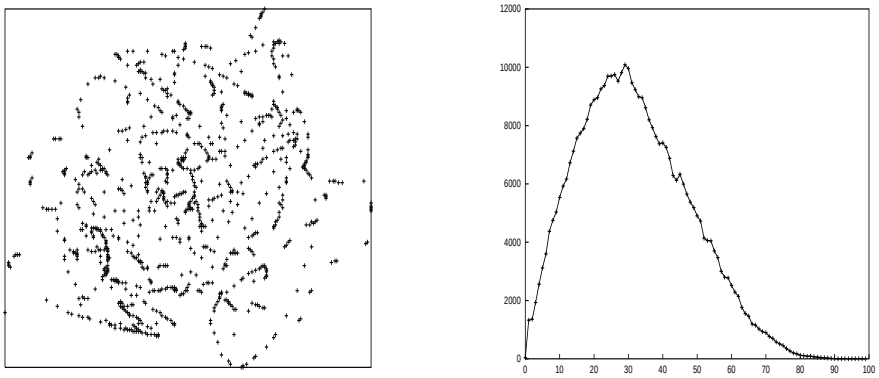

Figure 4: Random data set, Edge frequency histogram.

[4]Other than itself, of course, for which its distance is zero and is thus not counted

Figure 4 shows a random set of points and corresponding histogram. Note the similarity between the curve in Figure 4 and that in Figure 3. The presence of many small extrema in the curve indicates non-uniformity, but the fact that no large extrema exists indicates low clusterability.

Figure 5 shows a perfectly clusterable data set and the plot of its edge-frequency histogram. There is a large minimum at point 4 in the histogram and a large maximum at point 2. The separation between the internal and the external edges is clear; the internal edges are all represented in the initial spike of the histogram (buckets $0-4$), while the external edges are represented in the second spike (buckets $10-19$).

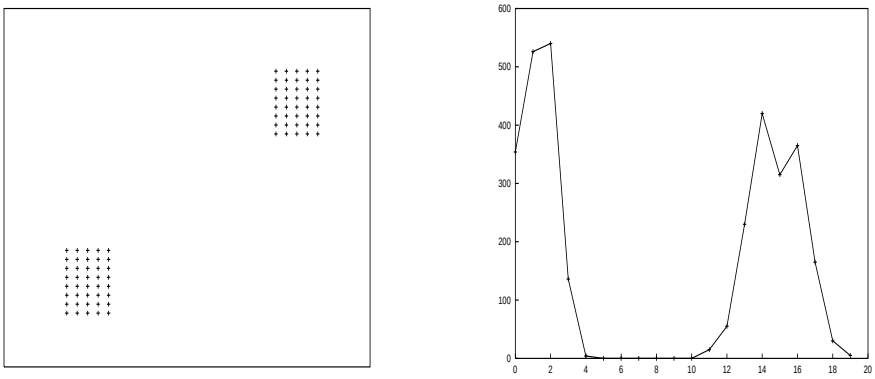

Figure 5: Two cluster data set, Edge frequency histogram.

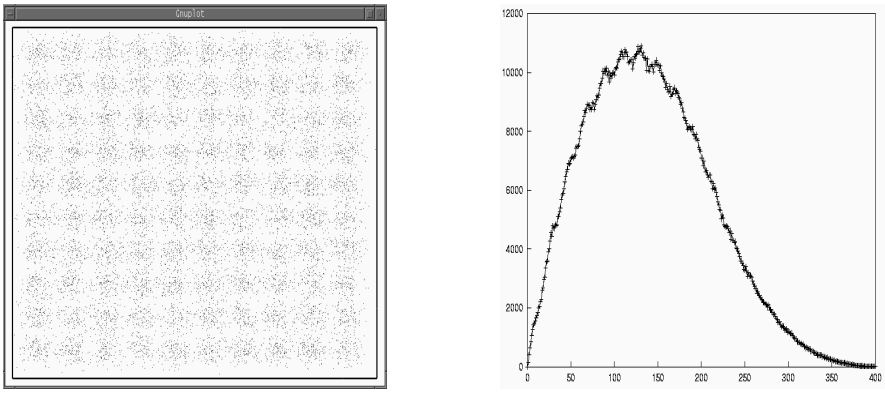

Figure 6: DS1 data set, Edge frequency histogram.

Figure 6 shows DS1 from BIRCH[21] and the plot of its edge-frequency histogram, generated from a random sample of the points. In this case, the uniformity of the data set is reflected in the similarity to the curve of the uniform set in Figure 3. The small degree of separation of the points is reflected in the jagged edges (small extrema) of the curve.

4.2 Implementation

Recall that there are two steps to the task at hand. First we generate the edge length frequency histogram for the data set in question and make an assessment of clusterability based on this histogram. In step two we determine a cutoff for internal and external edge lengths from the histogram generated in step one and then determine centroids using graph-theoretic concepts. Each of these steps is discussed in more detail here.

4.2.1 Histogram Generation

The histogram we use keeps track of the frequency of occurrence of values among the set of all-pairs distances between points (*edges* as defined in Section 3). In general for a data set of size n there are exactly $\frac{n^2-n}{2}$

edges in the corresponding graph. Obviously quadratic run times are impractical for large data sets and so in such cases a random sample is used. Much literature exists on the effect of validity and expected results of choice of sample size and we will not discuss it here [10, 11]. In practice, we have found that sample sizes under 10% have been sufficient for large data sets. In the discussion that follows we will assume that a data set refers to either an entire data set or a pre-chosen sample.

The only input parameter then is the number of buckets. Our experiments indicate that for any data set there exists a number of buckets b such that setting the number of buckets $> b$ will only result in an increased number of empty buckets. Very few buckets will result in a very coarse approximation of the distribution, while we get more precise answers for smaller bucket sizes (i.e., for more buckets). In our experience beyond a certain point the results are more or less insensitive to the number of buckets. Thus the choice of number of buckets is easy to make, it just need be sufficiently large.

Suppose then that there is a data set $\mathcal{D}$ and b buckets. We first compute the $\frac{|\mathcal{D}|^2-|\mathcal{D}|}{2}$ edge lengths $E_{u,v}$, determining the minimum and maximum values for $E_{u,v}$ as we go. Let us refer to these values as E_{min} and E_{max}, respectively. The bucket boundaries are determined by normalizing the range of b to the length of the interval $(E_{min} \ldots E_{max})$. The list of values is scanned, and a count is kept for each bucket.

In the second step, we scan the histogram and search for the existence of extreme points. A clusterability determination is made using the guidelines discussed in Section 4.1, and a suitable cutoff bucket c is chosen based on the existence of an initial large maximum followed by an initial large minimum (a spike, as in Figure 5) in the histogram. If no spike is found then the cutoff value must be approximated (see the experiments section for examples).

4.2.2 Seed Centroid Determination

We then construct $\mathcal{G}$ and determine connected components by performing a *depth-first* search on $\mathcal{G}$ with the added restriction that two nodes are adjacent if the bucket index that their edge falls into is $\leq c$. Note that this is identical to performing the search on $\mathcal{G}'$ and restricting adjacency to those nodes with + edges. The centroids of these connected point sets are the centroids in the output set.

The clusterability information referred to in step two is the quantity and type of extreme points in the histogram curve. The presence of several maxima followed by minima with a large span in between (spikes) indicates high clusterability, since it demonstrates concentrations of different types of edge lengths. The lack of any large gaps between maximum and minimum values indicates uniformity, whereas several small extrema indicates a small degree of separability.

There are several means of approximating c in the absence of a pronounced first spike. One method is to take c as some fraction of the bucket that represents the highest maximum value. Another method is to fix c as some small percentage of the total number of buckets. Finally we also envision using our techniques in conjunction with a visualization tool that allows the user to interactively choose the cut-off threshold and see the effect on the cluster seeds and number of clusters. Furthermore, in high dimensional spaces our approach can be used to identify projected cluster seeds in smaller dimensional subspaces [1, 2, 15].

Input:	a data set $\mathcal{D}$ and number of buckets b
Output:	a clusterability classification and initial set of centroids

Step 1:	find all edge lengths including E_{min} & E_{max}
	for each length
	  find normalized index
	  increment bucket counter for index

Step 2:	scan histogram find or approximate c output clusterability generate adjacency matrix find connected components for each connected component output centroid

Figure 7: Clusterability and Seed Generation Algorithm

We now give a high-level description of the complete algorithm (see Figure 4.2.2. We take as input a sample data set $\mathcal{D}$, along with the number of buckets to be used for the edge length histogram. In $O(n^2)$ time, where n is the number of points in the data sample $math\mathcal{D}$, we compute all pairwise edge lengths. We then normalize the lengths using the maximum E_{max} and minimum E_{min} edge found. The final task in step 1 is to construct the edge histogram.

In step 2 we scan the histogram looking for a good cut-off point c. The best scenario is for perfectly clusterable datasets, as shown in figure 5, where we find two peaks in the edge distributions, one corresponding to the internal edges and the other to the external edges. For other non-perfectly clustered datasets we have to approximate c as discussed above.

Once the cut-off has been chosen, we use it to obtain the signed graph or the graph of internal edges and find its connected components. The cluster seed is the centroid of all the points in a connected component.

5 Experimental Results

In this section we present the results of our experiments using the graph-based centroid detection algorithm. We begin with a description of the experiments performed and then give details of our results. In all experiments, Euclidean distance was the metric used.

The experiments were designed to show two things: *performance* in which we demonstrate the effect of dimensionality increase on running time and *accuracy* in which we show the results of centroid detection on easily visualizable (2 and 3-d) data sets. For small data sets, (< 3000) points, the entire data set is used. For larger data sets a random sample is used. The experiments were performed on a PentiumII 233 processor with 64M of memory, running Linux 2.0.36.

For the performance evaluation (Figure 8), we use the DS1 data set from BIRCH, as well as a data set containing roughly 500 values of Dow and Nasdaq closing prices. In order to increase the dimensionality of each data set, the first column of values is duplicated several times at the end of each record. So for example a file with 100 dimensions created from a 2-dimensional file will have each record with the first value, the second value, and then 98 copies of the first value.

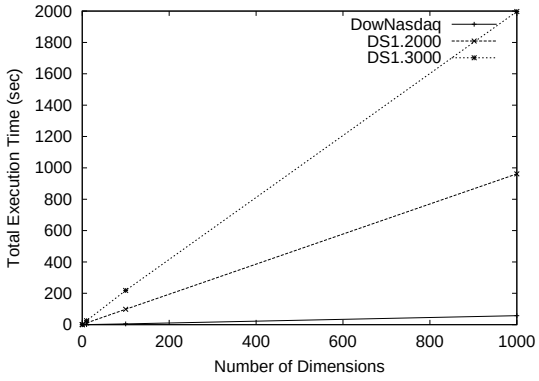

Figure 8: Effect of dimensionality on runtime.

For the accuracy evaluation (Figures 9- 15), we show the points of the original data set, with our detected centroids as darker, enlarged points. The histogram plot for each data set is also shown for evaluation. Figures 9- 11 show closing values for the S&P500 and the Dow, the S&P500 and the Nasdaq, and the Dow and the Nasdaq indices. Figure 12 shows the DS2 data set from BIRCH, and Figure 13 shows a 3-dimensional version of DS2 made using the procedure described above. Figure 14 shows a set of ellipsoid clusters drawn free-hand and Figure 15 shows a 3-dimensional version of the data set in Figure 14.

Performance Evaluation The effect on runtime for sample sizes of 2000 and 3000 points, respectively for DS1, and the entire data set (roughly 500) for DowNasdaq is shown. In the plots, the x-axis represents the dimension of the test file and the y-axis shows the run time. As can be seen from the plots, we achieve linear performance as dimensionality increases.

Accuracy Evaluation Figures 9- 15 show the edge length frequency histograms and detected centroids for several types of data sets. For all examples a bold, vertical line in the histogram indicates the chosen cutoff point c. In all but the final two sets (Figures 14- 15) c is approximated.

6 Conclusions and Future Work

We have presented a framework and methodology for determining *clusterability* of a data set, loosely based upon the ideas presented in [16]. Our work is based on an examination and graph-based traversal of the frequency histogram of edge-lengths between points in the data set. Part of the output of our system is an estimate of the value for k, which is required input for any variation of the most popular class of distance-based clustering methods. In addition to k we generate a set of seed centroids that can be used as intelligent starting points for any k-means style algorithm. The sensitivity of our algorithm is tunable, and so multiple centroids can be generated for algorithms that can exploit them, e.g. [12]. Our technique can be used in conjunction with a visualization tool that allows the user to interactively choose the cut-off threshold and see the effect on the cluster seeds and number of clusters. Furthermore, in high dimensional spaces our approach can be used to identify projected cluster seeds in smaller dimensional subspaces [1, 2, 15].

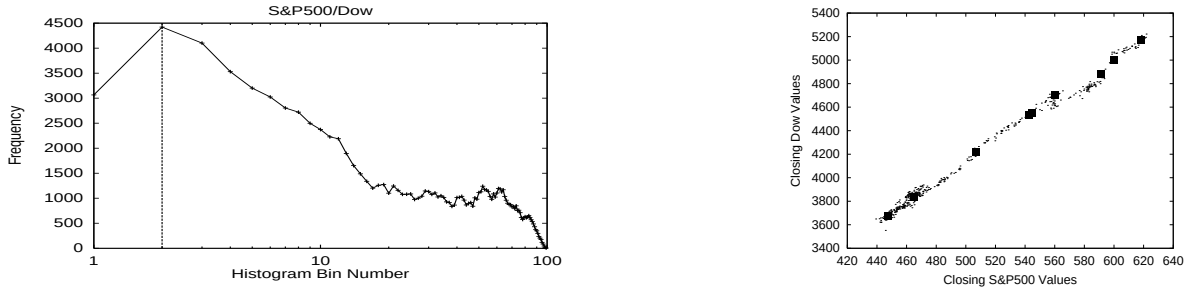

Figure 9: S&P500/Dow data set: $c = 2$, $k = 9$.

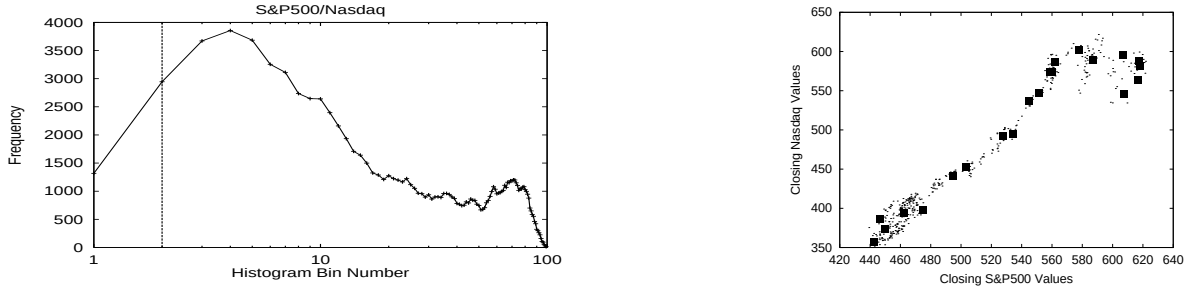

Figure 10: S&P500/Nasdaq data set: $c = 2$, $k = 21$.

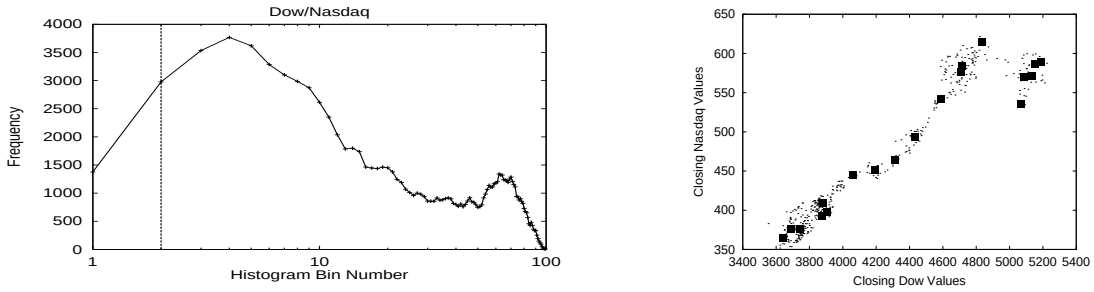

Figure 11: Dow/Nasdaq data set: $c = 2$, $k = 19$.

Several possibilities for improvement of this system exist. The most striking shortcomings are the sensitivity to noise, the need for sampling, and the need for automated detection of a reasonable cutoff point for estimating pseudo-internal and pseudo-external edges. Eventually, the determination of a representative set of centroids may even serve as an intelligent means of sampling a large data set. We also plan to test several k-means type and distance based algorithms by first seeding them with our cluster centers and to evaluate the resulting improvement.

We have only scratched the surface of the information that is available from the edge length histogram. Further analysis of the buckets beyond any initial spike will more than likely provide a wealth of as of yet untapped information, especially with respect to non-uniformity among cluster sizes (detection of *proper* clusterings as opposed to *perfect* ones).

We believe the framework to be a useful basis for the classification of data sets in terms of their clusterability and that this may eventually serve as a substitute for clustering altogether. Our group is currently working on a means of assigning an *index of clusterability* to a data set. Given a data set, a value in the range $[0 \ldots 1]$ would be generated, where *uniform* data sets would be assigned a value of 0 and *perfectly clusterable* data sets would be assigned a value of 1. Clusterability detection is an important, difficult and as of yet largely unexplored problem space.

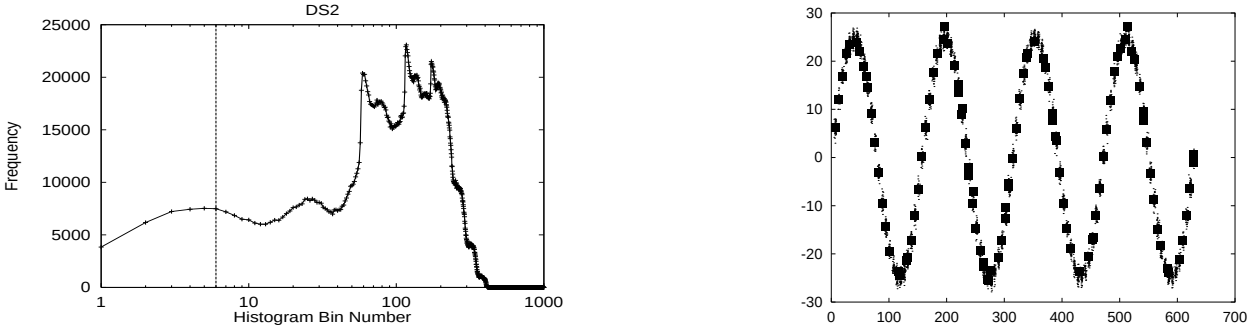

Figure 12: DS2 data set: $c = 6$, $k = 99$.

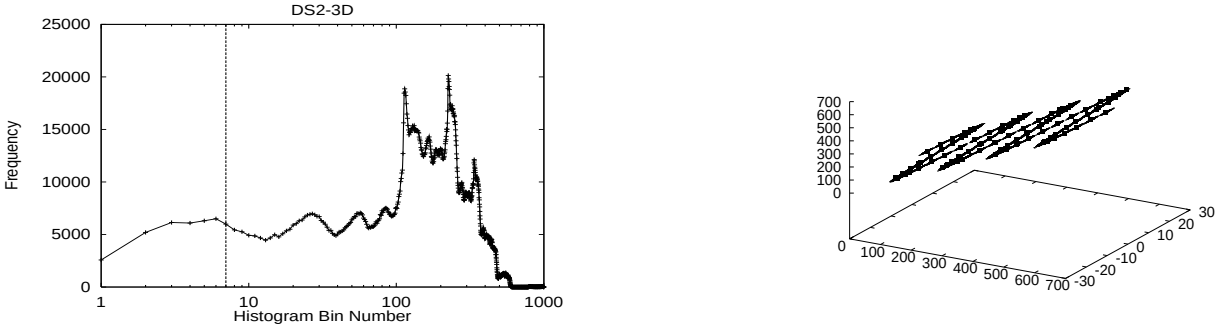

Figure 13: DS2.3d data set: $c = 7$, $k = 99$.

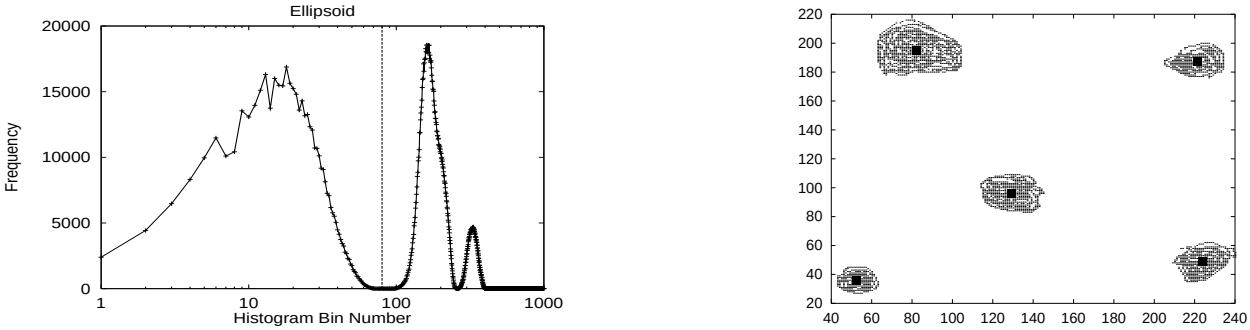

Figure 14: Ellipsoid data set: $c = 80$, $k = 5$.

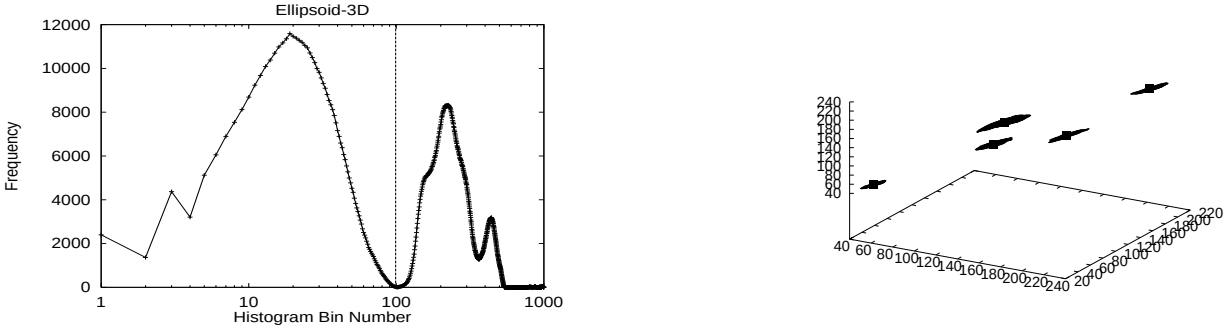

Figure 15: 3D-Ellipsoid data set: $c = 99$, $k = 5$.

References

[1] Charu C. Aggarwal and Philip S. Yu. Finding generalized projected clusters in high dimensional spaces. In *ACM SIGMOD International Conference on Management of Data*, May 2000.

[2] Rakesh Agrawal, Johannes Gehrke, and Dimitrios Gunopulos andPrabhakar Raghavan. Automatic subspace clustering of high-dimensional data for data mining applications. In *ACM SIGMOD Conference on Management of Data*, 1998.

[3] P. S. Bradley and Usama Fayyad. Refining initial points for k-means clustering. In *Proceedings of the International Conference on Machine Learning*, 1998.

[4] P. S. Bradley, Usama Fayyad, and Cory Reina. Scaling clustering algorithms to large databases. In *Proceedings of the International Conference on Knowledge Discovery in Databases*, 1998.

[5] Gary Chartrand. *Introductory Graph Theory*. Dover, 1975.

[6] Thomas H. Cormen, Cherles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. McGraw Hill, 1990.

[7] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[8] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Conference on Knowledge Discovery in Databases*, 1996.

[9] Usama Fayyad, Cory Reina, and P. S. Bradley. Initialization of iterative refinement clustering algorithms. In *Proceedings of the International Conference on Knowledge Discovery in Databases*, 1998.

[10] Phillip B. Gibbons and Yossi Matias. New sampling-based summary statistics for improving approximate query answers. In *ACM SIGMOD International Conference on Management of Data*, June 1998.

[11] Phillip B. Gibbons, Yossi Matias, and Viswanath Poosala. Fast incremental maintenance of approximate histograms. In *23rd International Conference on Very Large Data Bases*, August 1997.

[12] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: an efficient clustering algorithm for large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, 1998.

[13] Eui-Hong (Sam) Han, George Karypis, Vipin Kumar, and Bamshad Mobasher. Clustering in a high-dimensional space using hypergraph models. available at: www.cs.umn.edu/~karypis, 1998.

[14] J. Han and M. Kamber. Morgan Kaufmann Publishers, San Francisco, CA, 2001.

[15] A. Hinneburg, C. C. Aggarwal, and D. A. Keim. What is the nearest neighbor in high dimensional spaces? In *International Conference on Very Large Data Bases*, September 2000.

[16] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[17] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: and Introduction to Cluster Analysis*. John Wiley and Sons, 1990.

[18] Erwin Kreyszig. *Elementary Functional Analysis with Applications*. Wiley, 1978.

[19] Raymond T. Ng and Jiawei Han. Efficient and effective clustering methods for spatial data mining. In *Proceedings of the 20th VLDB Conference*, 1994.

[20] Gholamhosein Sheikholeslami, Surojit Chatterjee, and Aidong Zhang. Wavecluster: A multi-resolution clustering approach for very large spatial databases. In *Proceedings of the 24th VLDB Conference*, 1998.

[21] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases. In *ACM SIGMOD Conference on Management of Data*, 1996.