ClusterMap: Labeling Clusters in Large Datasets via Visualization

Keke Chen Ling Liu
College of Computing, Georgia Institute of Technology
801 Atlantic Dr.
Atlanta, GA 30329
1-404-385-2030
{kekechen, lingliu}@cc.gatech.edu

ABSTRACT
With the rapid increase of data in many areas, clustering on large datasets has become an important problem in data analysis. Since cluster analysis is a highly iterative process, cluster analysis on large datasets prefers short iteration on a relatively small representative set. Thus, a two-phase framework "sampling/summarization – iterative cluster analysis" is often applied in practice. Since the clustering result only labels the small representative set, there are problems with extending the result to the entire large dataset, which are almost ignored by the traditional clustering research. This extending is often named as labeling process. Labeling irregular shaped clusters, distinguishing outliers and extending cluster boundary are the main problems in this stage. We address these problems and propose a visualization-based approach to dealing with them precisely. This approach partially involves human into the process of defining and refining the structure "ClusterMap". Based on this structure, the ClusterMap algorithm scans the large dataset to adapt the boundary extension and generate the cluster labels for the entire dataset. Experimental result shows that ClusterMap can preserve cluster quality considerably with low computational cost, compared to the distance-comparison-based labeling algorithms.

Categories and Subject Descriptors
I.5.3 [**Clustering**]: Labeling algorithms for large datasets. H.1.2 [**User/Machine Systems**]: Human factors in clustering large datasets

General Terms
Algorithms, Human Factors

Keywords
Data Clustering, Cluster Labeling, Cluster Visualization, Human Factors in Clustering

CIKM'04, November 8-13, 2004, Washington, DC, USA

1. INTRODUCTION
Over the past decade, large datasets have been collected or produced in many application domains, such as bioinformatics, physics, geology, and marketing, and some have reached the level of terabytes or petabytes [22]. Therefore, there is a growing demand for efficient and precise clustering techniques that can adapt to the large datasets.

Several clustering algorithms have aimed at processing the entire dataset in linear or near linear time, for example, WaveCluster [14], DBSCAN [10], and DENCLUE [11]. However, there are some drawbacks with these approaches. First, the cluster analysis process often requires multiple runs of the clustering algorithms to find the optimal partitioning scheme. Even though a clustering algorithm has linear computational complexity, running such algorithm on a large dataset multiple times is still intolerable for many users. Second, existing clustering algorithms often work efficiently in finding clusters in spherical or elongated shapes but they cannot handle the arbitrarily cluster shapes very well, nor validate them effectively [13]. Some algorithms, such as OPTICS [15], try to find the arbitrarily shaped clusters, but their non-linear complexity often makes them only applicable to small or medium datasets.

Bearing the above problems in mind, a number of approaches were proposed to perform clustering algorithms on the sample datasets or data summaries instead of the entire large dataset. For example, CURE [2] applies random sampling to get the sample data and then runs a hierarchical clustering algorithm on the sample data. BIRCH [19] summarizes the entire dataset into a CF-tree and then runs a hierarchical clustering algorithm on the CF-tree. This "sampling/summarization – iterative cluster analysis" framework has been commonly recognized as a practical way in large-scale cluster analysis. Since the size of dataset is reduced with the sampling/summarization techniques, any typical clustering algorithms and cluster validation techniques that have acceptable non-linear computational complexity can be applied in cluster analysis. Therefore, clustering large dataset is seemly promisingly solved with this framework. However, the previous research on clustering almost ignored the following problem – how is the clustering result applied to the *entire* large dataset?

1.1 Labeling Clusters in Large Datasets: the Problem
One of the main problems with the two-phase framework is the gap between the clustering result of the representative dataset and

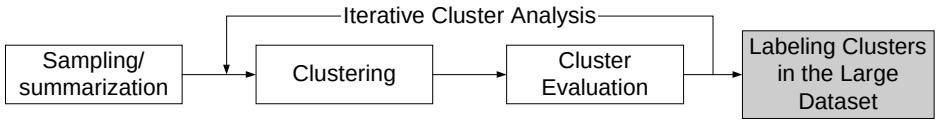

Figure 1-1. Cluster analysis and labeling

the requirement of retrieving cluster labels for the entire large dataset. Some typical questions asked by the applications are 1) what is the cluster label for the particular data item? and 2) what are the data items that belong to the particular cluster? Therefore, when the queries involve the items that are not in the representative dataset, we also need to deal with them without introducing inconsistency to the clustering structure.

Traditionally, the post-clustering stage is named labeling process. However, labeling is often ignored by the clustering research. Part of the reason is the clustering problem itself is still not well-solved. With the emerging of more and more effective clustering algorithms, which are typically not in linear complexity, the post-clustering stage will become critical to large dataset, especially when the points at the cluster boundary area are significant to the applications. Sampling or summarization theory makes the representative set "representative" enough to the significant clustering structure. However, when the sample size is only about 1% or even less than 1% of the original dataset, it also becomes necessary to listen to the opinions of the majority. A foreseeable problem in labeling large amount of data is that the cluster boundary will be extended more or less by incorporating the labeled data points. Boundary extension might result in the connection of different clusters and thus we may need to merge them. Since the boundary is extending, the outliers around the boundary should also be treated carefully. Figure 1-2 and 1-3 demonstrate the boundary extension problem.

The problems caused by cluster boundary extension are hard to handle mainly due to the difficulty in describing the boundary precisely in multidimensional space (>3D). The traditional labeling algorithms [2, 19] are based on the very rough description of cluster boundary, namely, a centroid or a set of representative boundary points. A typical labeling algorithm assigns each new data item to a cluster that has centroid or one representative boundary point closest to the date item. Centroid-based labeling uses cluster center only to represent a cluster. Representative-point-based labeling uses representative points on cluster boundary to describe the cluster, which is better than centroid-based labeling since it provides more information about the boundary. Obviously, precise description of boundary depends on the number of representative points in terms of different cluster shapes, which could be large for irregular shapes. While representative points are employed to describe boundary, it is also not easy to describe the boundary extension when more and more items are labeled. Furthermore, neither representative-point-based nor centroid-based labeling can deal with outliers satisfactorily. Figure 1-4 shows the lack of ability to dealing with the outliers when representative-point-based labeling is applied.

One important metric in evaluating labeling is *precision* that measures the consistency between the labeling result and the naturally extended clustering structure on the large dataset. In other words, this reflects how precisely the questions 1) and 2) get answered with the labeling result. Any rough labeling technique makes the meaningful intermediate clustering result inapplicable to the entire large dataset. Another important metric is the *complexity* of algorithm. Since labeling deals with the entire large dataset, it has to be linear or at least near-linear in time complexity.

To sum up, existing labeling algorithms have the following difficulties when applied to large datasets:

- Identifying irregularly shaped clusters is a hard problem in clustering, and it is also a challenge labeling clusters in large dataset.
- With the existing centroid-based or boundary-point-based labeling algorithms, it is difficult to identify the cluster outliers.
- If precise boundary description is considered, when the size of dataset becomes very large compared to the representative dataset, the boundary extension problem should be handled carefully.

To solve the above problems, we also need to keep in mind that

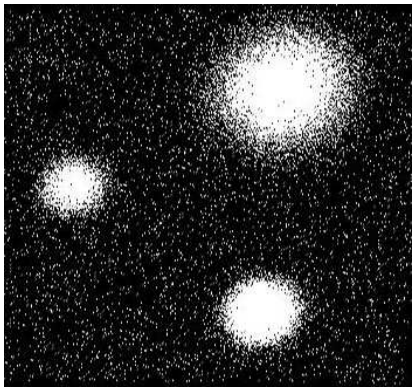

Figure 1-2. Cluster boundary in a small representative set (data points are white)

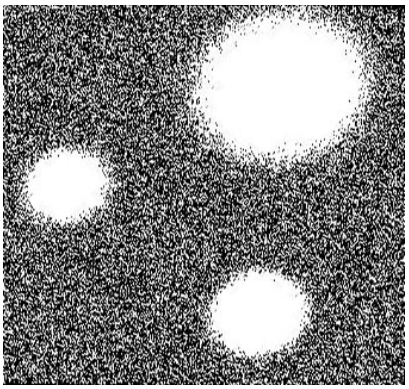

Figure 1-3. Cluster boundary is extended in the entire large dataset

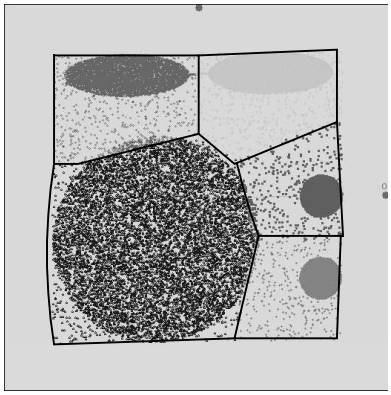

Figure 1-4. Outliers are labeled as the members of the nearby clusters.

the algorithm should be still in linear or near-linear complexity with the improved precision.

1.2 Our Approach

In this paper, we propose a new labeling approach, ClusterMap, which extends VISTA cluster rendering system [1, 20, 21] for labeling large datasets. The VISTA approach has shown several unique features: 1) identifying the irregular cluster shapes effectively via visualization; 2) validating and refining any algorithmic results visually to improve the cluster quality; 3) incorporating the domain knowledge conveniently; 4) having a flexible framework to incorporate algorithms and human interaction into the cluster analysis process. The visualization-based ClusterMap labeling makes a couple of new contributions to the post-clustering stage:

- We are the first to address the importance of labeling for cluster analysis of large datasets, and the problems with the distance-based labeling algorithms.
- We introduce visualization into the labeling stage, which is intuitive and flexible. ClusterMap is based on VISTA cluster rendering result and thus it can describe the arbitrary cluster shapes precisely, which reduces the labeling error caused by imprecise cluster representations.
- We develop a two-step ClusterMap processing algorithm, which eliminates the influence of boundary extension and allows the users to interactively examine the clustering structure for entire dataset. The "snapshots" of ClusterMap also enable the user to monitor the evolving of cluster structure caused by incorporating the processed items.

The rest of this paper is organized as follows. The subsection 1.3 gives some related work. Section 2 briefly describes VISTA system and its extension for large datasets. We then present the visualization-based labeling algorithm ClusterMap in section 3. Section 4 reports the experimental result, demonstrating the high quality of cluster preserving with low complexity by using ClusterMap. Finally, we conclude our work in section 5.

1.3 Related Work

A general cluster analysis framework is described in the review paper [9] of clustering techniques, which shows that cluster analysis is usually an iterative process, and the user always prefers faster algorithms or short response time. Thus, a common problem with large-scale clustering is the long response time. Even a fast algorithm running in linear time would let the user waiting for a while in an iterative cycle. CURE [2] and BIRCH [19] employ the "sampling/summarization – clustering" framework to deal with the large-scale clustering problem. The two-phase framework also facilitates the incorporation of other tools, such as VISTA cluster rendering system [1], for better understanding and refining of arbitrarily shaped clusters.

Dealing with arbitrarily shaped clusters is well-recognized as a hard problem in clustering research community. Several clustering algorithms have aimed at this particular problem, such as, CURE [2], CHAMELEON [17], DBSCAN [11], DBCLASD [16], WaveCluster [14], DENCLUE [12] and so on. But they were only reported effective in low dimensional dataset or in small/medium datasets. In conclusion, the automatic algorithms can deal with the arbitrarily shaped clusters to some extent, but the results are quite limited.

A semi-automatic algorithm OPTICS [15], which derives from DBSCAN [11] algorithm shows visualization can be very useful in cluster analysis. However, OPTICS is not applicable for large datasets unless applying the multi-phase framework. Other visualization systems, such as HD-Eye system [23], which are also limited by the size of the dataset, will need the labeling step when applied to large datasets.

Existing cluster representations can be classified into four categories: centroid-based, boundary-point-based (representative-point-based), classification-tree-based and rule-based representations [9]. Since the classification-tree-based and rule-based methods are equivalent (each path in the classification tree can be represented as a rule) and inapplicable in many situations, they are not widely used in practice. Using centroid to represent a cluster is the most popular scheme, since many algorithms produce only centroids for clusters. Obviously, it works only for the clusters having compact spherical shapes. Representative-point-based approach works better than centroids since it describes the clusters in more detail. But how to define the representative points precisely for arbitrarily shaped clusters is as difficult as the clustering problem.

2. EXTENDING THE VISUAL FRAMEWORK FOR LARGE DATASETS

The VISTA visual framework [20, 21] has shown that interactive visual cluster rendering can be very effective in identifying irregular cluster shapes and validating/refining the algorithmic results. We extend the visual framework to allow processing large datasets under the framework of "sampling – visual clustering – labeling with clusters". The extended visual framework is like Figure 2-1.

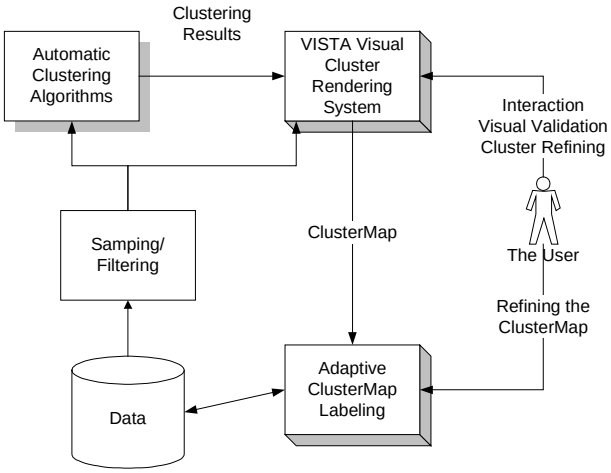

Figure 2-1. The extended visual framework

The large dataset is first sampled to get a subset in manageable size, which is then used as an input to the selected automatic clustering algorithms and the VISTA system. The algorithmic clustering result provides helpful information in visual cluster rendering process. The user interacts with the VISTA visual cluster rendering system to find the satisfactory visualization, which visualizes the clusters in well-separated areas. Since human vision is very sensible to the gap between point clouds, which imply the boundary of clusters, the interactive rendering works very well in defining vague boundary or irregular cluster shapes.

A ClusterMap is then defined on the satisfactory cluster visualization and used as the initial pattern in ClusterMap labeling. Finally, the labeling process will adapt the boundary extension and cluster refining in one pass through the entire dataset. An additional pass might be needed to reorganize the entire datasets for fast processing of queries. To further observe the small clusters that may be omitted in sampling process, the filtering component filters out the labeled outliers and performs sampling/visual rendering on the sampled outliers again. We will also briefly discuss small cluster processing in section 3.

2.1 VISTA System

To understand the ClusterMap algorithm better, we need to briefly review the VISTA system. The main problem in cluster visualization is cluster preserving, e.g. visualizing multi-dimensional datasets in 2D/3D visual space, while preserving the cluster structure. The past research and practice showed that preserving cluster structure precisely in static visualization, if not impossible, is very difficult and computationally costly [3,4,5,6,12]. A more practical way is to allow the user to interactively explore the dataset [3] to distinguish the unpreserved cluster structure, such as cluster overlapping, broken clusters and fake clusters (outliers in the original space are mapped to the same visual area). VISTA visual cluster rendering system [1, 20, 21] has shown how the interactions can be applied to find clusters. The visualization model is the core of the system. It uses a linear (or affine) mapping [18] – α-mapping with normalization to avoid the breaking of clusters after the *k*-dimensional to 2D space mapping. The interactive operations are used to find the visible "gaps" which help to discriminate the possible cluster overlapping. By finding the gaps and investigating the raw labels provided by automatic clustering algorithms, it is easy to identify the overlapping and fake clusters. The experiments have shown that visual cluster rendering can improve the understanding of clusters, validate and refine the algorithmic clustering result effectively.

In current version of VISTA, the system processes the Euclidean datasets only (where the distance/similarity function is defined by Euclidean distance) since they are the most common datasets in the applications. Therefore, by default, we refer to Euclidean datasets in the following discussion.

2.2 VISTA Mappings

The ClusterMap labeling is tightly related to the VISTA visualization model. The VISTA visualization model consists of two linear mappings – max-min normalization followed by α-mapping.

Max-min normalization is used to normalize the columns in the datasets in order to eliminate the predominant effect of large-valued columns. Max-min normalization with bounds [min, max] scales value *v* to [-1, 1] as follows:

$$v' = 2 * (v - \min)/(\max - \min) - 1, \quad (1)$$

v is the original value and *v'* is the normalized value.

α-mapping maps *k*-D points to 2D visual space while providing the convenience of visual parameter tuning. We describe α-mapping as follows. Let a 2D point $Q(x, y)$ represent the image of a *k*-dimensional (*k*-D) max-min normalized data point $P(x_1,\ldots x_i,\ldots,x_k)$, $x_i \in [-1, 1]$ in 2D space. $Q(x, y)$ is determined by the average of the vector sum of *k* vectors $\vec{S}_i \cdot x_i$, where $\vec{S}_i = (\cos(\theta_i), \sin(\theta_i))$, $i = 1..k$, $\theta_i \in [0, 2\pi]$ are the star coordinates [6] that represent the *k* dimensions on 2D visual space.

α-mapping: $Q(x, y)$ is determined by (2)

$$\{x, y\} = \left\{ (c/k)\sum_{i=1}^{k} \alpha_i x_i \cos(\theta_i) - x_0, \quad (c/k)\sum_{i=1}^{k} \alpha_i x_i \sin(\theta_i) - y_0 \right\}$$

α_i ($i = 1,2,\ldots k$, $-1 \le \alpha_i \le 1$) in the definition are the dimension adjustment parameters, one for each of the *k* dimensions. θ_i is set to $2\pi/i$ initially and can be adjusted either, but usually it is not necessary. (x_0, y_0) is the center of the display area and *c* is the scaling of the radius of display area. α-mapping is a linear mapping which ensures the visible gaps in 2D visualization are also the gaps in k-D space.

3. CLUSTERMAP ALGORITHM

3.1 The Basic ClusterMap Algorithm

ClusterMap is a convenient extending of VISTA cluster rendering system. When visual cluster rendering produces satisfactory visualization, we can set the boundary of a cluster by drawing a boundary to enclose it. Each cluster is assigned with a unique cluster identifier. After the cluster regions are marked, the entire display area can be saved (represented) as a 2D byte array (Figure 3-1). Each cell in the 2D array is labeled by an identifier – a cluster ID (>0) if it is in cluster region, or the outlier ID (=0), otherwise. Since the size of array is restricted by the screen size, we do not need a lot of space to save it. For example, the display area is only about 688*688 pixels on 1024*768 screen, slightly larger for higher resolution, but always bounded by a few mega pixels. As shown in Figure 3-1, the Cluster Map array is often a sparse matrix too, which can be stored more space-efficiently if necessary. Figure 3-2 is a visually defined ClusterMap of the 4D "iris" dataset. The boundaries of cluster C1, C2 and C3 were defined interactively.

In addition to store the 2D array, we need to save the mapping parameters for the labeling purpose. The parameters include:

- The max-min bounds of each column: $Cmax_j$ and $Cmin_j$. There are *k* pairs of such parameters totally, where *k* is the dimensionality of the dataset.
- The center of the visualization, e.g. (x_0, y_0)
- The *k* α parameters: $(\alpha_1, \alpha_2,\ldots, \alpha_k)$
- The scaling factor *c*
- The angles of the coordinates, $(\theta_1, \theta_2,\ldots, \theta_k)$

ClusterMap representation has several advantages. First, in most situations, the ClusterMap provides more details than the centroid-based or representative-point-based cluster representation, thus it is possible to preserve the precision of intermediate clustering result in labeling phase. Second, the cluster boundaries can be adjusted conveniently to adapt to any special situations or to incorporate domain knowledge as the way we use the VISTA system. Third, with ClusterMap representation

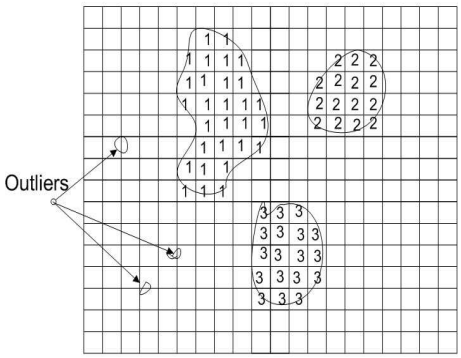

Figure 3-1. Illustration of basic ClusterMap, which has three clusters defined

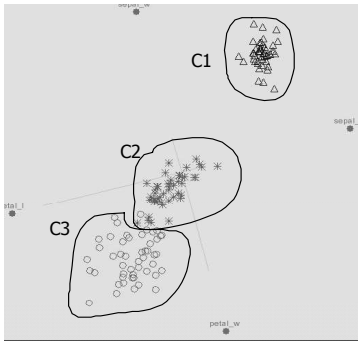

Figure 3-2. ClusterMap of the 4D "iris" dataset.

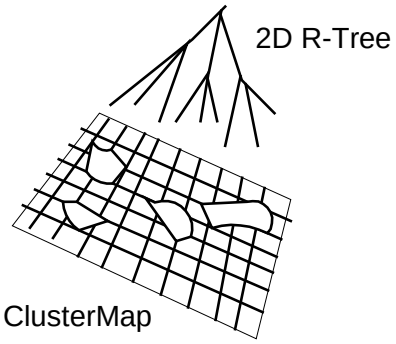

Figure 3-3. Build 2D R-tree over ClusterMap grid for fast retrieval

the outliers can be better distinguished. We shall see ClusterMap can also adapt the extension of cluster boundary in the following section.

The best way to discuss the features of the ClusterMap labeling is to compare it with the two typical cluster-labeling methods we have mentioned: Centroid-Based labeling (CBL) and Representative-Point-Based labeling (RPBL). To label a point, CBL compares the distances of this point to the centroids of clusters. The point is labeled with the cluster ID of the nearest centroid. RPBL utilizes the representative points generated by clustering process or fetched from the clustering results. It looks up the nearest neighbour of this point among all representative points and labels the point with the cluster ID of the nearest representative point. Both algorithms are kind of rough in describing the clusters and have difficulty in describing the arbitrarily shaped clusters or distinguishing outliers.

In comparison, the ClusterMap labeling is intuitively better and very straightforward. Having the ClusterMap loaded into the memory, each item in the entire large dataset is scanned and mapped onto one ClusterMap cell. The mapping follows the same mapping model used in the visual rendering system, which applies the max-min normalization first, and then followed by α–mapping. Suppose the raw dataset is stored in form of $N \times k$ matrix, where N is the number of rows and k is the number of columns. We rewrite the formulas as follows:

$$\textit{Normalization}: \; x'_{ij} = \pi_j * (x_{ij} - Cmin_j) - 1, \qquad (3)$$
$$\pi_j = 2/(Cmax_j - Cmin_j)$$

$$\alpha\_mapping : x_i = \sum_{j=1}^{k} px_j * x'_{ij} - x_0, y_i = \sum_{j=1}^{k} py_j * x'_{ij} - y_0 \qquad (4)$$

where $px_j = c * \alpha_j * \cos(\theta_j)/k$, $py_j = c * \alpha_j * \sin(\theta_j)/k$ and π_j can be pre-computed, and other parameters are the same as defined before.

Concretely, the algorithm reads the i-th item ($x_{i1} \ldots x_{ik}$) from the k-D raw dataset, normalizes and maps it with formulas (3) and (4) to a 2D cell coordinate (x_i, y_i). Reading the value stored in the cell (x_i, y_i), we gets the cluster ID label, which will be either 0 for outliers, or a positive integer for a cluster. For fast processing cluster related queries, we can also create block files to store the cluster members and build 2D R-tree index over the grid (Figure 3-3).

Given the formulas (3) and (4), we can roughly estimate the cost of ClusterMap labeling. We count the number of necessary multiplication to estimate the cost, for example, one k-D Euclidean distance calculation costs k multiplication. Map reading and parameter reading cost constant time. For each item in the dataset, the max-min normalization costs k multiplication with formula (3). The α-mapping function costs k multiplication respectively to calculate x, y coordinates with formula (4). Locating the cell in ClusterMap to get the corresponding cluster ID costs constant time. Hence, the total cost for the entire dataset is $3kN$, where N is the number of rows in the dataset.

Table 1: Cost estimation of the three algorithms.

k	Dimensionality
N	Total rows in raw dataset
n	The number of clusters
m	The number of representative points for each cluster
$f1$	The cost of ClusterMap, $3kN$
$f2$	The cost of CURE RPBL, $nmkN > f2 > log_2(nm) * kN$
$f3$	The cost of CBL, $nkN \geq f3 > log_2(n) * kN$

When kd-tree [25] or other multi-dimensional tree is used to organize the representative points or centroids, we get near-optimal complexity for the distance-comparison based labeling algorithms. The cost to find the nearest neighbour point in kd-tree is at least $log_2(nm)$ distance calculation for RPBL and at least $log_2(n)$ for CBL, where n and m as defined in table 2. For a typical RPBL as reported in the CURE paper, only when the number of representative points is greater than 10 ($m>=10$), the representative-point method can represent clusters roughly for regular non-spherical cluster shapes (mainly, the elongated shapes), and the more irregular the cluster shape the more representative points needed to describe the shape. Thus, the cost of RPBL will be at least $4kN$, even higher than ClusterMap.

So all the three algorithms are in O(N) complexity, which is ideal for processing large datasets. The main advantage of ClusterMap labeling is the precision of clustering structure preserved for the entire dataset. We will show in experiments how much the

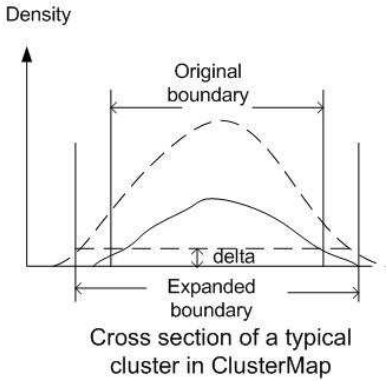

Figure 3-4. Boundary extension with the increase of dataset size

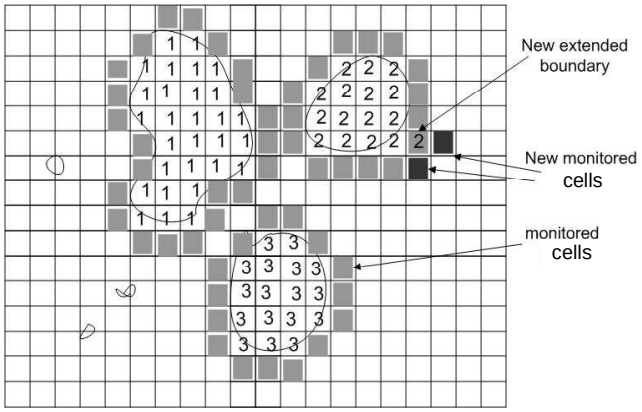

Figure 3-5. Illustration of monitored area where ε=1

ClusterMap labeling can reduce the error rate caused by irregular shape clusters, outliers, and boundary extension, compared to the other two algorithms.

3.2 Adaptive ClusterMap for Boundary Extension

In the basic ClusterMap labeling, we assume the cluster boundary (or a relaxed boundary as Figure 3-2 shows) defined on the sample set will not change a lot in labeling. However, with the number of labeled items increasing, the boundary will often extend more or less if the boundary is defined tightly and precisely on the sample set. An example has been shown in Figure 1-2 and 1-3. The boundary extension may also result in merging of two close clusters or enclosing the nearby outliers into the clusters. Figure 3-4 sketches that the possible extension happens in ClusterMap with the attending of labeled items.

Boundary extension is maintained by monitoring the point density around the boundary area. We have the initial boundary defined in VISTA at the beginning. We name the cells within the cluster boundary as the "cluster cells" and the cells around the boundary area as the "boundary cells". The initial boundary cells are precisely defined in a short distance $\boldsymbol{\varepsilon}$ away from the boundary defined by VISTA. All non-cluster cells are "outlier cells" including the boundary cells. We define *the density of a cell* on map as the number of items having been mapped to this cell. The density of boundary cells should be monitored in order to make decision on boundary extension. A threshold density of cell, $\boldsymbol{\delta}$, is defined as two times of the average density of outlier cells. If the density of a boundary cell grows to δ with the attending of labeled items, the boundary cell is turned to a cluster cell, resulting in the extension of boundary. The non-cluster cells within the ε-distance from the old boundary cell then become the new boundary cells. Since the boundary is on the 2D cells, we can use cell as the basic distance unit and "city block" distance [26] as the distance function to define ε-distance. ε is often a small number, for example, 1 or 2 "city block" distance from the current boundary. δ is growing as well as the density in non-cluster area is growing so the measuring of boundary extension keeps consistent with the density of non-cluster area.

To support the above adaptive algorithm, we need to extend the basic structure of ClusterMap. First of all, for each cell, we need one more field to indicate if it is a monitored non-cluster cell. We also need to keep track of the number of points falling onto each cell, which is saved in a *density map*. Since the average noise level will inevitably rise with the increase of labeled items, δ should be periodically updated according to the average noise level. The following algorithm briefly describes the adaptive ClusterMap labeling.

---

1. ε-distance (boundary i) $\rightarrow$ boundary (i),
   g_o = # of outlier cells,
   n_o = # of outlier points,
   $\delta = 2*n_o / g_o$
2. a new point $\rightarrow \eta$ with formulas (3) and (4)
3. density(η) ++,
4. if $\eta \notin$ cluster
5. then if $\eta \in$ boundary (i) and density (η) > δ
6. then $\eta \rightarrow$ cluster(i);
7. for $\eta' \in \varepsilon$-distance(η) $\cap$ $\eta' \notin$ cluster (i)
8. $\eta' \rightarrow$ boundary (i)
9. $g_o = g_o - 1$
10. $n_o = n_o$ – density(η)
    else
11. $n_o = n_o + 1$
12. $\delta = 2*n_o / g_o$
13. Goto 2. and repeat until processed all points

Algorithm 1. The first scan over the entire dataset to adapt the boundary extension

Boundary extension can behave abnormally, which might be caused by inappropriate initial cluster boundary. During the first scan, "snapshots" of ClusterMap are saved and visually monitored by the user. Snapshots are a series of evolving ClusterMaps, which incorporate the boundary extension, saved at some time interval during labeling. The user can terminate the labeling process early if the snapshots show bad trends, for example, the cluster boundary becomes vague soon after labeling some points and we cannot distinguish the clusters any more. Normal boundary extension should be slow and uniformly distributed around the boundary. After first scan, the user can also make decision on merging clusters or creating new clusters for small emerging clusters, which are often ignored by the

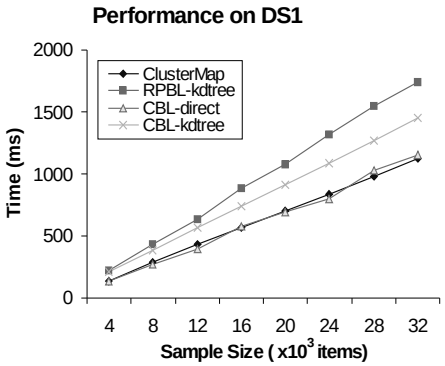

Figure 4-1. Cost on DS1

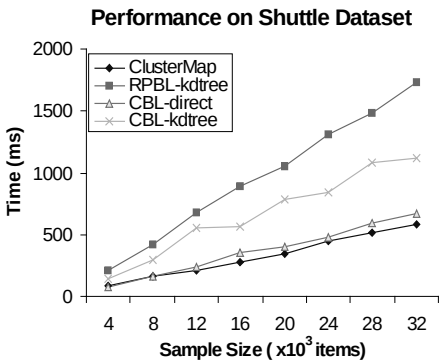

Figure 4-2. Cost on Shuttle data

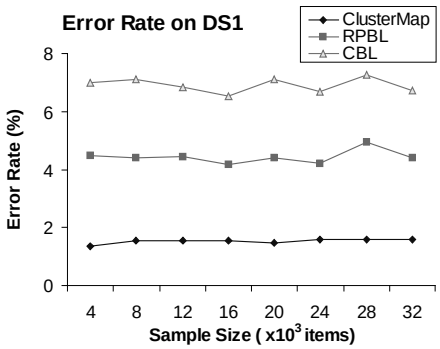

Figure 4-3. Error rate on DS1

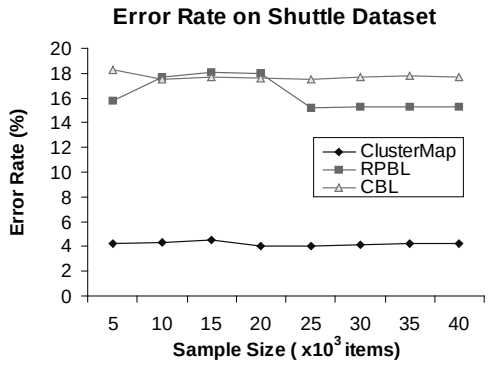

Figure 4-4. Error rate on Shuttle

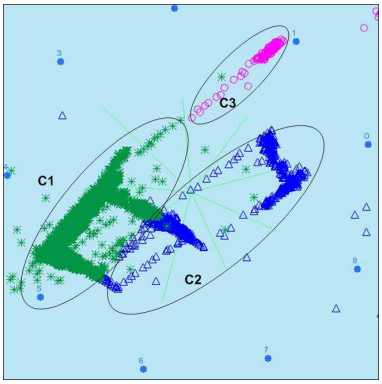

Figure 4-5. RPBL on Shuttle

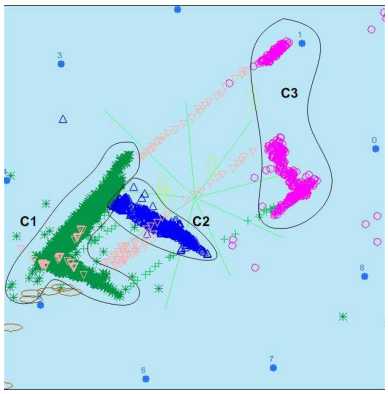

Figure 4-6. Documented clusters on Shuttle

sampling/summarization process. Since this paper is not focused on the interaction part, we will not describe the operations in details.

You may also concern that data ordering will affect boundary extension. For example, a sequence of data is mapped to a focused boundary area at early stage thus extends the boundary, but later on no more points are mapped to that area. As a result, this area is falsely extended as a part of cluster. This problem can be avoided by randomizing the processing sequence.

The first scan generated an adapted ClusterMap. The second scan can be performed to build up a R-tree index on the map for the items as Figure 3-3 shows. Clustering applications could involve similarity search [24] and this additional index structure will be helpful for such applications. However, we can also skip the second scan and leave the computational cost at answering ad-hoc queries if the queries come infrequently.

3.3 Observe and Label the Small Clusters

Sampling can possibly cause the loss of small clusters [2]. In this section we briefly describe how to capture the small clusters in an iterative method.

During the first scan of ClusterMap labeling, we can check if there are small clusters emerging from the outliers on the "snapshots". If there are small clusters emerging, we can use the following filtering method to observe the emerging small clusters in details. Since after the first scan clusters and the extended boundary are well defined, we can subtract them from the large dataset and observe the rest outlier dataset only. If the outlier dataset is still large, it is sampled and observed in VISTA cluster rendering system. Since the size of outlier dataset should be much smaller than that of the original dataset, the sample will show more details of the outliers and the smaller clusters could be well preserved after sampling. Again, the observed clusters are marked in another ClusterMap. We can repeat this process until the size of the outliers become uninterestingly small. Due to the space limitation, we do not discuss the details or experiments about this problem in the paper.

4. EXPERIMENTS

This section presents two sets of experiments. The first set of experiments shows ClusterMap can handle the labeling of outliers and irregular shaped clusters much better. The second set of experiments demonstrates the advantage of ClusterMap in dealing with cluster boundary extension for large datasets.

4.1 Datasets and Experiment Setup

Three datasets are used for the two sets of the experiments. One is the simulated dataset DS1 used in CURE. DS1 is a 2D dataset having five regular clusters, including three spherical clusters, two connected elliptic clusters, and many outliers. In our experiments, DS1 is used to evaluate the effect of outliers on the labeling algorithms. The second dataset is a real dataset – Shuttle dataset (STATLOG version, test dataset). It is a 9-dimensional dataset with very irregular cluster distribution. There are seven clusters in this dataset, among which one is very large with approximately 80% of data items, and two are moderately large with approximately 15% and 5% of data items, respectively. The others are tiny. Shuttle dataset is used to evaluate the effect of clustering

result on the labeling process. These two datasets have original labels, so we can calculate exact error rate with the original labels and generated labels. The third dataset is a simulated 5-dimensional large dataset LDS in size of $6*10^7$ items (up to 4G), which contains five spherical clusters and some outliers. Figure 4-7 shows a 10K sample set visualized with VISTA system. The first two datasets should show how ClusterMap avoids the common problems of the traditional algorithms. The third dataset is used to show that adaptive ClusterMap works well with very large datasets.

The three algorithms, CBL, RPBL, and ClusterMap are implemented in C++. RPBL is based on the CURE clustering algorithm, which was known as the best RPBL adapted for non-spherical cluster. We run CURE clustering with the following parameters: the number of representative points is 20 and alpha (shrink factor) is set to 0.5 as suggested. We also use ANN (Approximate Nearest Neighbour) C++ library from University of Maryland at College Park to construct *kd*-tree for RPBL and CBL.

4.2 Outliers and Irregular Cluster Shapes

We have discussed three different algorithms: Representative-Point Based Labeling (RPBL), Centroid-Based Labeling (CBL), and ClusterMap. In this section we study the performance and error rates of ClusterMap experimentally, compared to the two commonly used algorithms RPBL and CBL.

We run VISTA to get the ClusterMap in the resolution of 688*688. The cost to rebuild a ClusterMap structure in memory is about 340~360ms. In contrast the cost to build *kd*-tree is about 1~2ms. Both DS1 and shuttle datasets show the cost estimation is appropriate – all three are linear and the basic ClusterMap is slightly fast (Figure 4-1 and 4-2). Linear complexity is good enough for a one-time building process.

The DS1 dataset is used to show the effect of outliers to the algorithms. The error rate of RPBL is on average 4.5% over DS1; the error rate of the CBL has the error rate of 6.8%; while the error rate of ClusterMap has only about 1.5%, much lower than the other two. ClusterMap also shows a more stable error rate. From the visualization of the labeling results, we observed that CBL suffers from the large circle cluster and outliers. Most RPBL errors come from the outliers. Since RPBL result is better than CBL we only show the visualization of RPBL results (Figure 1-4). The visualization of RPBL labeling shows that the outliers are labeled as the member of the nearby cluster, which is the main reason why RPBL yields higher error rate.

Shuttle dataset has very irregular clusters. With a small number of "landmark points" [20], we can easily and correctly define the clusters with VISTA. We use the shuttle dataset to show the limitation of the popular RPBL algorithm to the very irregular cluster shapes. Figure 4-4 shows that the error rate of RPBL raises to the level, similar to that of CBL (about 17%). ClusterMap labeling keeps consistent with the VISTA cluster rendering result and thus has only 4.2% of incorrect labels, much lower than the other two algorithms.

In the visualization of RPBL on 10000-item subset (Figure 4-5), we can see it divides the original cluster c3 into two parts and cannot discriminate c1 and c2. We give the correct centroids for CBL, however, centroids simply cannot represent the irregular cluster distribution, which makes the error rate very high.

4.3 Boundary Extension on Large Dataset

This experiment on the large dataset LDS mainly shows the scalability of adaptive ClusterMap algorithms and the advantage of ClusterMap in handling boundary extension. LDS simulates 5 spherical clusters, two big and three small clusters, as well as ~1% background noise. LDS is well designed so that we can predefine the control labels for entire dataset with small error. The actual boundaries are larger than the visible ones on the sample set as shown in Figure 4-7. We did not consider the merging between the close clusters, such as B and C, since it is not considered by RPBL and CBL. The progressive result of performance and error rate is shown in Figure 4-8 and 4-9. The performance curve shows that the adaptive ClusterMap costs slightly less than the other algorithms. Since the clusters are spherical, the labeling error of RPBL and CBL comes from the improper boundary description and the outliers. In this regular cluster distribution, representative points and centroids have the similar effect in cluster description, and thus, RPBL and CBL have almost the same result as Figure 4-9 shows. The error of ClusterMap mainly comes from the minor imprecision around the boundary are. It is normal that the boundary produced by adaptive ClusterMap might not be perfect in such a noised dataset. However, one of the ClusterMap's advantages is that the users can always check and tune the ClusterMap in such situations.

5. CONCLUSION

With the emerging of more and more large datasets, clustering

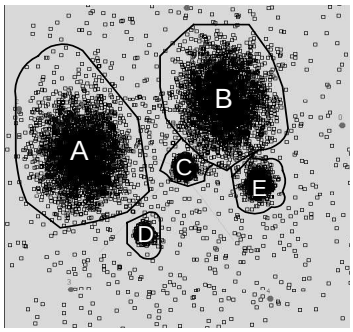

Figure 4-7. Visualization of 10K samples of LDS

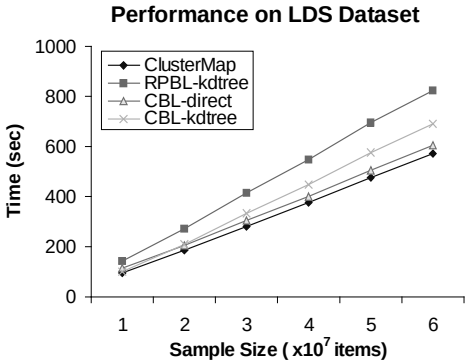

Figure 4-8. Performance on LDS

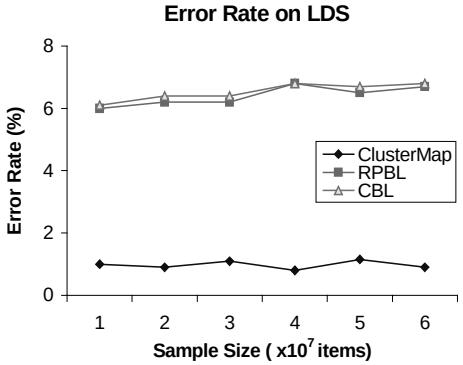

Figure 4-9. Error rate on LDS

large datasets has become a major problem in data analysis. The three-phase framework "sampling/summarization – iterative cluster analysis – labeling large datasets with clusters" provides a feasible solution for cluster analysis on large datasets. Labeling large datasets with clusters bridges the effective cluster analysis to the large dataset. There are some difficult problems coming with the labeling problem - labeling arbitrarily shaped clusters, distinguishing outliers, and adapting the extension of cluster boundary. In this paper, we addressed these problems and propose a visual framework for large-scale cluster analysis, which combines VISTA visual cluster rendering system and ClusterMap labeling algorithm. The experimental result shows ClusterMap algorithm usually preserves more clustering structure than the existing methods.

It is well believed that human participation should help clustering problem. However, there are very few projects demonstrating the visualization power on clustering, and our work uniquely shows how human can help in preserving the clustering quality for large dataset.

6. ACKNOWLEDGMENTS

This research is partially supported by NSF CNS CCR, NSF ITR, DoE SciDAC, DARPA, CERCS Research Grant, IBM Faculty Award, IBM SUR grant, HP Equipment Grant, and LLNL LDRD.

Any opinions, findings, and conclusions or recommendations expressed in the project material are those of the authors and do not necessarily reflect the views of the sponsors.

7. REFERENCE

[1] Chen, K. and Liu, L. Cluster Rendering of Skewed Datasets via Visualization. *In Proceedings of ACM Symposium on Applied Computing 2003 (SAC03)*, Melborne, FL. 2003

[2] Guha, G., Rastogi, R., and Shim, K. CURE: An efficient clustering algorithm for large databases. *In Proceedings of the 1998 ACM SIGMOD*, Seattle, WA. 1998

[3] Keim, D. Visual Exploration of Large Data Sets. *Communications of the ACM*, August 2001, V. 44. No. 8

[4] Cook, D.R, Buja, A., Cabrea, J. and Hurley, H. Grand tour and projection pursuit. *Journal of Computational and Graphical Statistics*, 4(3) 155-172, 1995

[5] Yang, L. Interactive Exploration of Very Large Relational Datasets through 3D Dynamic Projections, *in Proc. of SIGKDD2000*, Boston, MA, 2000

[6] Kandogan, E. Visualizing Multi-dimensional Clusters, Trends, and Outliers using Star Coordinates, *in Proc. of SIGKDD2001*, San Francisco, CA, 2001.

[7] Jain, A. and Dubes, R. *Algorithms for Clustering Data*. Prentice hall, Englewood Cliffs, NJ, 1988

[8] Grinstein, G., Ankerst, M. and Keim, D. Visual Data Mining: Background, Applications, and Drug Discovery Applications, *Tutorial at ACM SIGKDD2002*, Edmonton, Canada. 2002

[9] Jain, A., Murty, M.N. and Flynn, P.J. Data Clustering: A Review. *ACM Computing Surveys*, 31(3), P264-323, 1999

[10] Ester, M., Kriegel, H., Sander, J. and Xu, X. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, *in Proc. of VLDB96*, Bombay, India, 1996

[11] Hinneburg, A. and Keim, D. An Efficient Approach to Clustering in Large Multimedia Databases with Noise, *in Proc. of KDD98*, NYC, NY, 1998

[12] Shneiderman, B. Inventing Discovery Tools: Combinning Information Visualization With Data Mining. *Information Visualization*, 1, p5-12, 2002

[13] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. Cluster Validity Methods: Part I&II, *SIGMOD Record*, Vol31, No.2&3, 2002

[14] Sheikholeslami, G., Chatterjee, S. and Zhang, A. Wavecluster: A multi-resolution clustering approach for very large spatial databases, *In Proc. VLDB98*, NYC, NY, 1998

[15] Ankerst, M., Breunig, M., Kriegel, H. and Sander, J. OPTICS: Ordering Points To Identify the Clustering Structure. *In Proc. of SIGMOD1999*, Philadelphia, PA, 1999

[16] Xu, X., Ester, M., Kriegel, H. and Sander, J. A Distribution-based Clustering Algorithm for Mining in Large Spatial Databases. *In Proc. of ICDE98*, Orlando, FL, 1998

[17] Karypis, G., Han, E. and Kumar, V. CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modelling. *IEEE Computer*, 32(8), 68-75, 1999

[18] Gallier, J. *Geometric methods and applications: for computer science and engineering*. Springer-Verlag, NY, 2001

[19] Zhang, T., Ramakrishnan, R. and Livny, M. BIRCH: An efficient data clustering method for very large databases, *In Proc. of SIGMOD96*, 103-114, Montreal, Canada, 1996

[20] Chen, K. and Liu, L. A visual framework invites human into clustering process. *In Proc. of Statistical and Scientific Database Management (SSDBM03)*, Boston, MA, 2003

[21] Chen, K. and Liu, L. Validating and Refining Clusters via Visual Rendering. *In Proc. of Intl. Conf. on Data Mining (ICDM03)*, Melborne, FL, 2003

[22] Huber, P. Massive data sets workshop: The morning after Massive Data Set, *National Academy Press*, 1996

[23] Hinneburg, A., Keim, D. and Wawryniuk, M. Visual Mining of High-dimensional data. *IEEE Computer Graphics and Applications*. 19(5), 1999

[24] Li, C., Chang, E., Garcia-Molina, H. and Wiederhold, G. Clustering for Approximate Similarity Search in High-Dimensional Spaces, *IEEE Trans. on Knowledge and Data Engineering*, 14(4), 792-808, 2002

[25] Freidman, J.H., Bentley, J.L., and Finkel, R.A. An algorithm for finding best matches in Logarithmic Expected Time. *ACM Trans. on Mathematical Software*. 3(3), 1977

[26] Sonka, M., Hlavac, V. and Boyle, R. *Image Processing, Analysis, and Machine Vision*. Brooks/Cole Publishing, Pacific Grove, CA, 1999