Efficient Hierarchical Clustering of Large Data Sets Using P-trees [1]

Anne Denton, Qiang Ding, William Perrizo
Department of Computer Science,
North Dakota State University
Fargo, ND 58105-5164, USA
{anne.denton, qiang.ding, william.perrizo}@ndsu.nodak.edu

Qin Ding
Department of Computer Science,
Penn State Harrisburg
Middletown, PA 17057, USA
qding@psu.edu

Abstract

Hierarchical clustering methods have attracted much attention by giving the user a maximum amount of flexibility. Rather than requiring parameter choices to be predetermined, the result represents all possible levels of granularity. In this paper a hierarchical method is introduced that is fundamentally related to partitioning methods, such as k-medoids and k-means, as well as to a density based method, namely center-defined DENCLUE. It is superior to both k-means and k-medoids in its reduction of outlier influence. Nevertheless it avoids both the time complexity of some partition-based algorithms and the storage requirements of density-based ones. An implementation is presented that is particularly suited to spatial-, stream-, and multimedia data, using P-trees[1] for efficient data storage and access.

1 INTRODUCTION

Many clustering algorithms require parameters to be chosen to determine the granularity of the result. Partitioning methods such as the k-means and k-medoids [1] algorithms require that the number of clusters, k, be specified. Density-based methods, e.g., DENCLUE [2] and DBScan [3], use input parameters that relate directly to cluster size rather than the number of clusters. Hierarchical methods [1, 4] avoid the need to specify either type of parameter and instead produce results in the form of tree structures that include all levels of granularity. When generalizing partitioning-based methods to hierarchical ones, the biggest challenge is the performance. Determining a k-medoids-based clustering for only one value of k has already got a prohibitively high time complexity for moderately sized data sets. Many solutions have been proposed [5, 6, 7], such as CLARA [1] and CLARANS [5], but an important fundamental issue remains, namely that the algorithm inherently depends on the combined choice of cluster centers. In this paper we show how this unfavorable dependence can be avoided. Our proposed solution is related to the density-based clustering algorithm DENCLUE [2] albeit with a different justification. Based on our definition of a cluster center, we define a hierarchy that naturally structures clusters at different levels. Moving up the hierarchy uniquely identifies each low-level cluster as part of a particular higher-level one. Low-level cluster centers provide starting points for the efficient evaluation of higher-level cluster centers. Birch [4] is another hierarchical algorithm that uses k-medoids related ideas without incurring the high time complexity. In Birch, data are broken up into local clustering features and then combined into CF Trees. In contrast to Birch we determine cluster centers that are defined globally. We represent data in the form of P-trees [10, 11, 12, 13, 14], which are efficient both in the storage requirements and the time complexity of computing global counts.

The rest of the paper is organized as follows. In section 2 we analyze the reasons for high time complexity of k-medoids-based algorithms and show how this complexity can be avoided using a modified concept of a cost function. In section 3 we introduce an algorithm to construct a hierarchy of cluster centers. In section 4 we discuss our implementation using P-Trees. Performance analysis is given in Section 5.

2 A LOOK AT ESTABLISHED ALGORITHMS

Partition-based and density-based algorithms are commonly seen as fundamentally and technically distinct. Work on combining both methods has focused on an applied rather than a fundamental level [8]. We will present three of the most popular algorithms from the two categories in a context that allows us to extract the essence of both. The existing algorithms we consider in detail are k-medoids [1] and k-means [9] as partitioning techniques, and the center-defined version of DENCLUE [2] as a density-based one. Since the k-medoids algorithm is commonly seen as producing a useful clustering, we start by reviewing its definition.

2.1 K-Medoid Clustering as a Search for Equilibrium

A good clustering in k-medoids is defined through the minimization of a cost function. The most common choice of the cost function is the sum of squared Euclidean distances between each data item and its closest

[1] Patents are pending on the P-tree technology. This work is partially supported by GSA Grant ACT#: K96130308.

cluster center. An alternative way of looking at this definition considers cluster centers with position $X^{(m)}$ as defining an energy landscape for the data points at locations $X^{(i)}$, where the sum is taken over points in the cluster defined by $X^{(m)}$

$$E(X^{(m)}) = \sum_{i=1}^{N} \sum_{j=1}^{d} (x_j^{(i)} - x_j^{(m)})^2$$

where N is the number of data points that are assumed to influence the cluster center. The location of the energy minimum is calculated as follows:

$$\frac{\partial}{\partial x_j^{(m)}} E(X^{(m)}) = -2 \sum_{i=1}^{N} (x_j^{(i)} - x_j^{(m)}) = 0$$

The minimum of the potential is therefore the mean of coordinates of the data points to which it is attracted.

$$x_j^{(m)} = \frac{1}{N} \sum_{i=1}^{N} x_j^{(i)}$$

This result illustrates a fundamental similarity between the k-medoids and the k-means algorithm. Similar to k-means, the k-medoids algorithm produces clusters where clusters centers are data points close to the mean. The main difference between two algorithms lies in the degree to which they explore the configuration space of cluster centers. Whereas k-medoids related techniques extensively search that space, k-means corresponds to a simple hill-climbing approach. Therefore the fundamental challenge of avoiding the complexity in k-medoids related techniques consists in eliminating the dependency of the definition of one cluster center on the position of all others. This dependency can be avoided by choosing a cutoff for the attractive potential of cluster centers that does not depend on cluster boundaries but on distance instead. A natural choice for such a potential is a Gaussian function.

Cluster centers are now determined as local minima in the potential landscape defined by a superposition of Gaussian functions for each data point. This approach is highly related to the density-based algorithm DENCLUE [2]. In DENCLUE the Gaussian function is one possible choice for a so-called influence function that is used to model the overall density. We will take over the term "influence function" used in DENCLUE noting that strictly our influence function is the negative of DENCLUE's.

We would not expect the cluster centers in this approach to be identical to the k-medoids ones because a slightly different problem is solved but both can be well motivated. A particular benefit of using an entirely distance-based influence function results from the fact that outliers have a less significant influence.

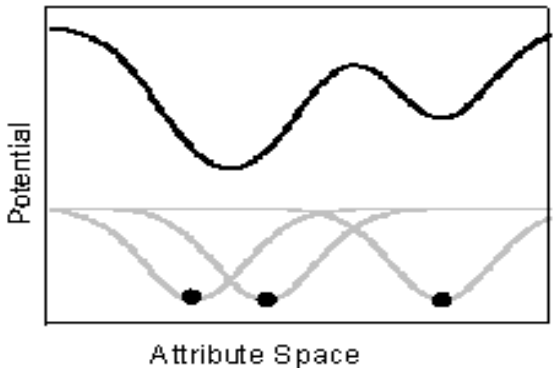

Figure 1: Energy landscape (black) and potential of individual data items (gray) for a Gaussian influence function

3 HIERARCHICAL ALGORITHM

We have now motivated the following procedure for determining cluster centers. The clustering process is viewed as a search for equilibrium of cluster centers in an energy landscape that is given by the sum of the Gaussian influences of all data points $X^{(i)}$

$$E(X) = -\sum_i e^{-\frac{(d(X, X^{(i)}))^2}{2\sigma^2}}$$

where the distance d is taken to be the Euclidean distance calculated in the d-dimensional space of all attributes

$$d(X, X^{(i)}) = \sqrt{\sum_{j=1}^{d} (x_j - x_j^{(i)})^2}$$

It is important to note that the minima of "potential energy" depend on the effective range of the interaction. This range is parameterized by the width of the Gaussian influence function, σ. This width specifies the range for which the potential approximates a quadratic function. It directly determines 2σ as the minimum cluster size.

Rather than treating σ as a constant that has to be chosen by the user, we iterate through a sequence of values. This is computationally inexpensive since we can use the configuration at an iteration i, σ_i, as starting point for the next minimization. Choosing 2σ to be smaller than the smallest distance between data points would ensure that every data point forms its own cluster. This is the common starting point for hierarchical agglomerative methods such as Agnes [1]. In practice it is sufficient to choose 2σ such that it is less than the smallest cluster size in which the user is likely to be interested. The cluster centers at this lowest level are considered the leaves of a tree, while branches will join at higher levels for increasing σ. Cluster boundaries are defined in such a way that data points are always grouped with the cluster centers they are closest to as in the k-means / k-medoids

algorithms. We will call the resulting structure the equilibrium tree of the data.

4 ALGORITHM

Treating clustering as a search for equilibrium cluster centers requires us to have a fast method of finding data points based on their feature attribute values. Density-based algorithms such as DENCLUE achieve this goal by saving data in a special data structure that allows referring to neighbors. We use a data structure, namely a Peano Count Tree (or P-tree) [10, 11, 12, 13, 14] that allows fast calculation of counts of data points based on their attribute values.

4.1 A Summary of P-Tree Features

Many types of data show continuity in dimensions that are not themselves used as data mining attributes. Spatial data that is mined independently of location will consist of large areas of similar attribute values. Data streams and many types of multimedia data, such as videos, show a similar continuity in their temporal dimension. Peano Count Trees are constructed from the sequences of individual bits, i.e., 8 P-trees are constructed for byte-valued data. Compression is achieved by eliminating nodes that consist entirely of 0- or 1-values. Two and more dimensional data is traversed in Peano order, i.e., recursive raster order. This ensures that continuity in all dimensions benefits compression equally. Counts are maintained for every quadrant. The P-tree for an 8-row-8-column bit-band is shown in Figure 2.

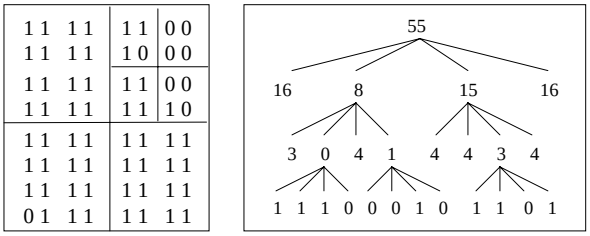

Figure 2. 8×8 image and its P-tree.

A major benefit of the P-tree structure lies in the fact that calculations can be done in the compressed form. This allows efficiently evaluating counts of attribute values or attribute ranges through an "AND" operation on all relevant P-trees.

4.2 Energy Evaluation using HOBBit Intervals

The first step in our algorithm consists in constructing the P-trees from the original data. Starting points for the minimization are selected randomly although other strategies would be possible. As minimization step we evaluate neighboring points, with a distance (step size) *s*, in the energy landscape. This requires the fast evaluation of a superposition of Gaussian functions. We use equality in the High Order Basic Bit or HOBBit distance [11] to intervalize the data. The HOBBit distance between two integer coordinates is equal to the number of digits by which they have to be right shifted to make them identical. The HOBBit distance of the binary numbers 11001 and 11011, for example, would be 2. The number of intervals that have distinct HOBBit distances is equal to the number of bits of the represented numbers. For more than one attribute, the HOBBit distance is defined as the maximum of the individual HOBBit distances of the attributes. The range of all data items with a HOBBit distance smaller than or equal to d_H corresponds to a d-dimensional hyper-cube. Counts within these hyper-cubes can be efficiently calculated by an "AND" operation on P-trees.

We start with 2σ being smaller than any cluster center we are interested in and then pick a potential cluster center to minimize its energy. The minimization is done in standard valley decent fashion. Further steps in the hierarchy use a larger σ with each of the lower level cluster centers as starting point for the minimization. If cluster centers move to the same minimum they are considered a single node of the hierarchy.

5 PERFORMANCE ANALYSIS

We tested the speed and effectiveness of our algorithm by comparing with the result of using k-means clustering. For storage requirements of P-trees we refer to [14]. We used synthetic data with 10% noise. The data was generated with no assumptions on continuity in the structural dimension (e.g., location for spatial data, time for multimedia data). Such continuity would significantly benefit from the use of P-tree methods. The speed demonstrated in this section can therefore be seen as an upper bound to the time complexity. Speed comparison was done on data with 3 attributes for a range of data set sizes. Figure 3 shows the time that is required to find the equilibrium location of one cluster center with an arbitrary starting location (leaf node) and with a starting location that is an equilibrium point for a smaller σ (internal node). We compare with the time required to find a k-means clustering for a particular k (k=2 in figure 3) divided by k. It can be seen that our algorithm shows the same or better scaling and approximately the same speed as k-means. Since k-means is known to be significantly faster than k-medoids-based algorithms, this performance can be considered highly competitive with partition-based methods.

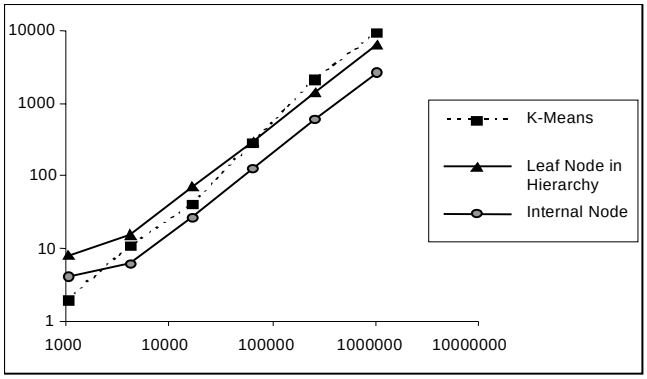

Figure 3: Speed comparison between our approach and k-means

To determine clustering effectiveness we used a small data set with two attributes and visually compared results with the histogram (Figure 4). Table 1 compares the cluster centers of k-means for k = 5 with those found by our algorithm. K-means results were significantly more influenced by the noise between the identifiable clusters than the results of our algorithm.

Table 1: Comparison of cluster centers for the data set of figure 3.

k-means (k=5)	x	11	4	25	4	23
	y	11	12	6	22	23
our algorithm	x	9	27	24	4	18
	y	11	22	6	21	25

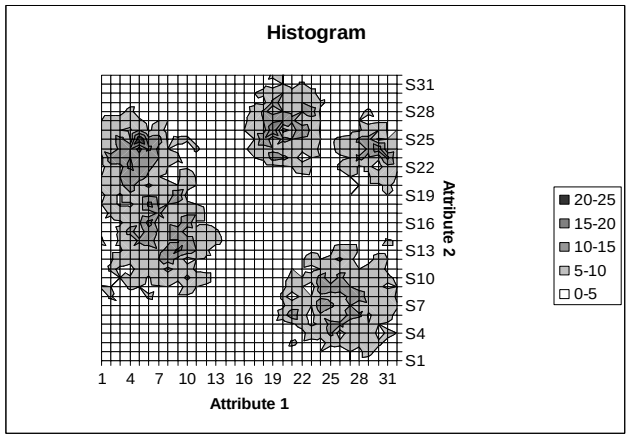

Figure 4: Histogram of values used to compare our approach with k-means

6 CONCLUSIONS

We have developed a hierarchical clustering algorithm that is based on some of the same premises as well-known partition- and density-based techniques. The time-complexity of k-medoids related algorithms is avoided in a systematic way and the influence of outliers is reduced. The hierarchical organization of data represents information at any desired level of granularity and relieves the user from the necessity of selecting parameters prior to clustering. Different levels in the hierarchy are efficiently calculated by using lower level solutions as starting points for the computation of higher-level cluster centers. We use the P-tree data structure for efficient storage and access of data. Comparison with k-means shows that we can achieve the benefits of improved outlier handling without sacrificing performance.

7 REFERENCES

[1] L. Kaufman and P.J. Rousseeuw, "Finding Groups in Data: An Introduction to Cluster Analysis", New York: John Wiley & Sons, 1990.

[2] A. Hinneburg and D. A. Keim, "An Efficient Approach to Clustering in Large Multimedia Databases with Noise", KDD'98, New York, Aug. 1998.

[3] M. Ester, H.-P. Kriegel, and J.Sander, "Spatial Data Mining: A Database Approach", SSD'97, Berlin, Germany, July 1997.

[4] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases", SIGMOD'96, Montreal, Canada, June 1996.

[5] R. Ng and J. Han, "Efficient and Effective Clustering Method for Spatial Data Mining", VLDB'94, Santiago, Chile, Sept. 1994.

[6] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification", SSD'95, Portland, ME, Aug. 1995.

[7] P. Bradley, U. Fayyard, and C. Reina, "Scaling Clustering Algorithms to Large Databases", KDD'98, New York, Aug. 1998.

[8] M. Dash, H. Liu, X. Xu, "1+1>2: Merging Distance and Density Based Clustering", DASFAA, 2001.

[9] J. MacQueen, "Some Methods for Classification and Aanalysis of Multivariate Observations", Proc. 5th Berkeley Symp. Math. Statist. Prob., 1:281-297, 1967.

[10] Qin Ding, Maleq Khan, Amalendu Roy, and William Perrizo, "P-tree Algebra", ACM Symposium on Applied Computing, Madrid, Spain, 2002.

[11] Maleq Khan, Qin Ding, William Perrizo, "K-Nearest Neighbor Classification of Spatial Data Streams using P-trees", PAKDD-2002, Taipei, Taiwan, May 2002.

[12] Qin Ding, Qiang Ding, William Perrizo, "Association Rule Mining on Remotely Sensed Images using P-trees", PAKDD-2002, Taipei, Taiwan, 2002.

[13] Qin Ding, William Perrizo, Qiang Ding, “On Mining Satellite and other RSI Data”, DMKD-2001, Santa Barbara, CA, 2001.

[14] A. Roy, “Implementation of Peano Count Tree and Fast P-tree Algebra”, M. S. thesis, North Dakota State University, 2001.