A Neighborhood-Based Clustering Algorithm*

Shuigeng Zhou[1], Yue Zhao[1], Jihong Guan[2], and Joshua Huang[3]

[1] Dept. of Computer Sci. and Eng., Fudan University, Shanghai 200433, China
{sgzhou, zhaoyue}@fudan.edu.cn

[2] Dept. of Computer Sci. and Eng., Tongji University, Shanghai 200092, China
jhguan@tongji.edu.cn

[3] E-Business Technology Institute, The University of Hong Kong, Hong Kong, China
jhuang@eti.hku.hk

Abstract. In this paper, we present a new clustering algorithm, NBC, *i.e.*, Neighborhood Based Clustering, which discovers clusters based on the neighborhood characteristics of data. The NBC algorithm has the following advantages: (1) NBC is effective in discovering clusters of arbitrary shape and different densities; (2) NBC needs fewer input parameters than the existing clustering algorithms; (3) NBC can cluster both large and high-dimensional databases efficiently.

1 Introduction

As one of the most important methods for knowledge discovery in databases (KDD), clustering is very useful in many data analysis scenarios, including data mining, document retrieval, image segmentation, and pattern classification [1]. Roughly speaking, the goal of a clustering algorithm is to group the objects of a database into a set of meaningful clusters each of which contains objects as similar as possible according to a certain criterion. Currently, mainly four types of clustering algorithms have been developed, including hierarchical, partitioning, density-based and grid-based algorithms.

With the fast development of data collection and data management technologies, the amount of data stored in various databases increases rapidly. Furthermore, more and more new types of data come into existence, such as image, CAD data, geographic data, and molecular biology data. The hugeness of data size and the variety of data types arise new and challenging requirements for clustering algorithms. Generally, a good clustering algorithm should be *Effective*(e.g. be able to discover clusters of arbitrary shape and different distributions), *Efficient*(e.g. be able to handle either very large databases or high-dimensional data-bases, and *Easy to use*(e.g. need no or few input parameters).

However, there are few current clustering algorithms can meet fully the 3-E criteria above-mentioned. In this paper, we present a new clustering algorithm:

* This work is supported by the Natural Science Foundation of China under grant No. 60373019 and 60496325, and partially supported by IBM-HKU Visiting Scholars Program.

T.B. Ho, D. Cheung, and H. Liu (Eds.): PAKDD 2005, LNAI 3518, pp. 361–371, 2005.

Neighborhood-Based Clustering algorithm (NBC in abbr.). The NBC algorithm uses the neighborhood relationship among data objects to build a neighborhood based clustering model to discover clusters. The core concept of NBC is the *Neighborhood Density Factor* (NDF in abbr.). NDF is a measurement of *relative local density*, which is quite different the *absolute global density* used in DBSCAN [2]. In this sense, NBC can still be classified into density based clustering algorithms. However, comparing with DBSCAN(the pioneer and representative of the existing density based clustering algorithms), NBC boasts of the following advantages:

- NBC can automatically discover clusters of arbitrary distribution, it can also recognize clusters of different local-densities and multi-granularities in one dataset, while DBSCAN uses global parameters, it can not distinguish small, close and dense clusters from large and sparse clusters. In this sense, NBC is closer to the *Effective* criterion than DBSCAN.

  To support this point, let us see a dataset sample shown in Fig. 4(a). In this dataset, there are totally five clusters, in which three are dense and close to each other (near the center of the figure) and the other two are much sparse and locate far away (locating near the upper-right angle and the upper-left angle of the figure respectively). Distance between any two of the three dense clusters is not larger than the distance between any two points in the two sparse clusters. With such a dataset, no matter what density threshold is taken, DBSCAN can not detect all the five clusters. In fact, when the density threshold is selected low, DBSCAN can find the two sparse clusters, but the three dense clusters are merged into one cluster; In contrast, when the threshold is set high, DBSCAN can find the three dense clusters, but all data points in the two sparse clusters are labelled as noise. However, NBC can easily find all the five clusters. We will give the clustering results in the performance evaluation section.
- NBC needs only one input parameter(the k value), while DBSCAN requires three input parameters(the k value, the radius of the neighborhood, and the density threshold). That is, NBC needs fewer input parameters than DBSCAN, so NBC is advantageous over DBSCAN in view of the *Easy to Use* criterion.
- NBC uses cell-based structure and VA file [3] to organize the targeted data, which makes it be efficient and scalable even for very large and high dimensional databases.

With all these advantages, we do not intend to replace the existing algorithms with NBC. Instead, we argue that NBC can be a good complement to the existing clustering methods.

2 A Novel Algorithm for Data Clustering

2.1 Basic Concepts

The key idea of neighborhood-based clustering is that: for each object p in a cluster, the number of objects whose k-nearest-neighborhood contains p should not less than the number of objects contained in p's k-nearest-neighborhood. In what follows, we give the formal definition of neighborhood-based cluster and its related concepts.

Given a dataset $D=\{d_1, d_2, \ldots, d_n\}$, p and q are two arbitrary objects in D. We use Euclidean distance to evaluate the distance between p and q, denoted as $dist(p, q)$. We will first give the definitions of k-nearest neighbors set and reverse k-nearest neighbors set. Although similar definitions were given in the literature, we put them here to facilitate the readers to understand our new algorithm.

Definition 1 (k-Nearest Neighbors Set, or simply kNN). The k-nearest neighbors set of p is the set of k ($k > 0$) nearest neighbors of p, denoted by kNN(p). In other words, kNN(p) is a set of objects in D such that
(a) $|k\mathrm{NN}(p)| = k$;
(b) $p \notin k\mathrm{NN}(p)$;
(c) Let o and o' be the k-th and the $(k+1)$-th nearest neighbors of p respectively, then $dist(p, o') \geq dist(p, o)$ holds.

Definition 2 (Reverse k-Nearest Neighbors Set, or simply R-kNN). The reverse k-nearest-neighbors set of p is the set of objects whose kNN contains p, denoted by R-kNN(p), which can be formally represented as

$$R\text{–}kNN(p) = \{q \in D | p \in kNN(q) \ and \ p \neq q\}. \quad (1)$$

Note that in the literature, reverse kNN is usually abbreviated as RNN. Here we use R-kNN rather than RNN because every RNN set is evaluated based on a certain k value.

kNN(p) and R-kNN(p) expose the relationship between object p and its neighbors in a two-way fashion. On one hand, kNN(p) describes who makes up of its own neighbors; On the other hand, R-kNN(p) indicates whose neighborhood p belongs to. This two-way description of the relationship between an arbitrary object and its neighborhood gives a clearer and more precise picture of its position in the dataset both locally and globally, which depends on the value of k, than simply using only kNN. In what follows, we will give the definitions of an object's neighborhood.

Definition 3 (r-Neighborhood, or simply rNB). Given a positive real number r, the neighborhood of p with regard to (abbreviated as w.r.t. in the rest part of this paper) r, denoted by rNB(p), is the set of objects that lie within the circle region with p as the center and r as the radius. That is,

$$rNB(p) = \{q \in D | dist(q, p) \leq r \ and \ q \neq p\}. \quad (2)$$

Definition 4 (k-Neighborhood, or simply kNB). For each object p in dataset D, $\exists o \in k\mathrm{NN}(p)$, $r' = dist(p, o)$ such that $\forall o' \in k\mathrm{NN}(p)$, $dist(p, o') \leq r'$. The k-neighborhood of p, written as kNB(p), is defined as r'NB(p), *i.e.*, kNB(p)= r'NB(p). We call kNB(p) as p's k-neighborhood w.r.t. kNN(p).

Definition 3 and Definition 4 define two different forms of neighborhood for a given object from two different angles: rNB(p) is defined by using an explicit radius; In contrast, kNB(p) is defined by using an implicit radius, which corresponds to the circle region covered by kNN(p). It is evident that $|k\text{NB}(p)| \geq k$ because there could be more than one object locating on the edge of the neighborhood (the circle). Accordingly, we define the reverse k-Neighborhood as follows.

Definition 5 (Reverse k-Neighborhood, or simply R-kNB). The reverse k-neighborhood of p is the set of objects whose kNB contains p, denoted by R-kNB(p), which can be formally written as

$$R\text{–}kNB(p) = \{q \in D | p \in kNB(q) \ and \ p \neq q\}. \tag{3}$$

Similarly, we have $|\text{R-}k\text{NB}(p)| \geq |\text{R-}k\text{NN}(p)|$.

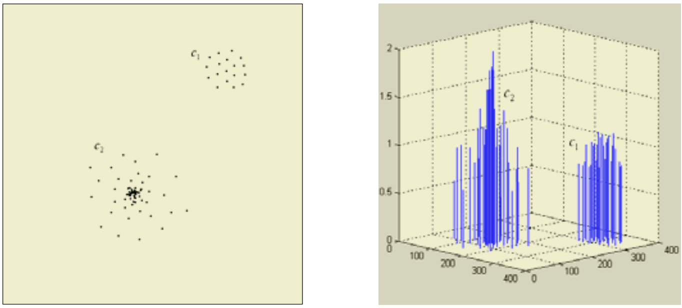
In a local sense, data points in a database can be exclusively classified into three types: *dense point*, *sparse point* and *even (distribution) point.* Intuitively, points within a cluster should be dense or even points; and points on the boundary area of a cluster are mostly sparse points. Outliers and noise are also sparse points. Currently, most density-based clustering algorithms (*e.g.* DBSCAN) use an intuitive and straightforward way to measure density, *i.e.*, a data object's density is the number of data objects contained in its neighborhood of a given radius. Obviously, this is a kind of absolute and global density. Such a density measurement makes DBSCAN unable to detect small, close and dense clusters from large and sparse clusters. In this paper, we propose a new measurement of density: *Neighborhood based Density Factor* (or simply NDF), which lays the foundation of our new clustering algorithm NBC.

Definition 6 (Neighborhood-based Density Factor, or simply NDF). The NDF of point p is evaluated as follows:

$$NDF(p) = \frac{|R\text{–}kNB(p)|}{|kNB(p)|}. \tag{4}$$

Then what is the implication of NDF? Let us check it. $|k\text{NB}(p)|$ is the number of objects contained in p's k-nearest neighborhood. For most data objects, this value is around k (According to Definition 4, it maybe a little greater, but not less than k). $|\text{R-}k\text{NB}(p)|$ is the number of objects contained in p's reverse k-nearest neighborhood, *i.e.*, the number of objects taking p as a member of their k-nearest neighborhoods. This value is quite discrepant for different data points. Intuitively, the larger $|\text{R-}k\text{NB}(p)|$ is, which implies that the more other objects take p as a member of their k-nearest neighborhoods, that is, the denser p's neighborhood is, or the larger NDF(p) is. In such a situation, NDF(p) > 1. For uniformly distributed points, if q is in kNB(p), then p is most possibly in kNB(q), therefore, kNB(p) $\approx$ R-kNB(p), that is NDF(p) ≈ 1. Thus, NDF is actually a measurement of the density of any data object's neighborhood, or data object's local density in *relative*(*not absolute*) sense. Furthermore, such a measurement is intuitive(easy understanding), simple(easy implementation) and effective(being able to find some cluster structures that DBSCAN can not detect).

To demonstrate the capability of NDF as a measurement of local density, we give an example in Fig. 1. Fig.1(a) is a dataset that contains two clusters C_1,



(a) Dataset

(b) NDF values of data points

Fig. 1. An illustration of NDF

C_2. Data in C_1 is uniformly distributed; and data in C_2 conforms to Gaussian distribution. Fig. 1(b) shows the NDF values of all data points in the dataset. As we can see, data points locate within cluster C_1 have NDF values approximately equal to 1, while data points locating on boundary of C_1 have smaller NDF values. For cluster C_2, the densest point is near the centroid of C_2, which has the largest NDF value, while other objects have smaller NDF values, and the further the points locate from the centroid, the smaller their NDF values are.

With NDF, in what follows, we give the definitions of three types of data points in local sense: *local event points*, *local dense points* and *local sparse points*.

Definition 7 (Local Dense Point, simply DP). Object p is a *local dense point* if its NDF(p) is greater than 1, also we call p a dense point w.r.t. kNB(p), denoted by DP w.r.t. kNB(p). The larger NDP(p) is, the denser p's k-neighborhood is.

Definition 8 (Local Sparse Point, simply SP). Object p is a *local sparse point* if its NDF(p) is less than 1, and we call p a sparse point w.r.t. kNB(p), denoted by SP w.r.t. kNB(p). The smaller NDP(p) is, the sparser p's k-neighborhood is.

Definition 9 (Local Even Point, simply EP). Object p is a local even point if its NDF(p) is equal (or approximately equal) to 1. We call p an even point w.r.t. kNB(p), denoted by EP w.r.t. kNB(p).

With the concepts defined above, in what follows, we introduce the concepts of neighborhood-based cluster. Our definition follows the way of DBSCAN.

Definition 10 (Directly neighborhood-based density reachable). Given two objects p and q in dataset D, p is *directly neighborhood-based density reachable* (*directly ND-reachable* in abbr.) from q w.r.t. k, if

(a) q is a DP or EP, and
(b) $p \in k\text{NB}(q)$.

Definition 11 (Neighborhood-based density reachable). Given two objects p and q in dataset D, p is *neighborhood-based density reachable* (*ND-reachable* in abbr.) from q w.r.t. k, if there is a chain of objects $p_1, \cdots, p_n, p_1 = p, p_n = q$ such that p_i is directly ND-reachable from p_{i+1} w.r.t. k.

According to Definition 11, if object p is directly ND-reachable from object q, p is surely ND-reachable from q.

Definition 12 (Neighborhood-based density connected). Given two objects p and q in a dataset D, p and q are *neighborhood-based density connected* (*ND-connected* in abbr.) w.r.t. k, if p is ND-reachable from q w.r.t. k or q is ND-reachable from p w.r.t. k or there is a third object o such that p and q are both ND-reachable from o w.r.t. k.

With the concepts above, now we are able to define the *neighborhood-based cluster* as follows.

Definition 13 (Neighborhood-based cluster). Given a dataset D, a cluster C w.r.t. k is a non-empty subset of D such that

(a) for two objects p and q in C, p and q are ND-connected w.r.t. k, and
(b) if $p \in C$ and q is ND-connected from p w.r.t. k, then $q \in C$.

The definition above guarantees that a cluster is the maximal set of ND-connected objects w.r.t. k.

2.2 The NBC Algorithm

The NBC algorithm consists of two major phases:

- *Evaluating NDF values.* We search kNB and R-kNB for each object in the target dataset, and then calculate its NDF.
- *Clustering the dataset.* Fetch an object p randomly, if p is a DP or EP (NDF(p)$\geq$1), then create a new cluster, denoted as p's cluster, and continue to find all other objects that are ND-reachable from p w.r.t. k, which involves all objects belonging to p's cluster. Otherwise, if p is a SP, then just put it aside temporarily, and continue to retrieve the next point to process. This work is recursively done until all clusters are discovered. More concretely, given a DP or EP p from the database, first finding the objects that are directly ND-reachable from p w.r.t. k. Objects in kNB are the first batch of such objects, which will be moved into p's cluster. Then finding the other objects directly ND-reachable from each DP or EP in p's cluster until there is no more object can be added into p's cluster. Second, from the rest of the dataset, fetching another DP or EP to build another cluster. When there is no more DP or EP to fetch to create clusters, the algorithm comes to an end. Points belonging to none cluster are noise or outliers. Fig.2 outlines the NBC algorithm in C pseudo-code.

Here, *Dataset* indicates the dataset clustered, k is the only input parameter used in NBC to evaluate kNB and R-kNB. The value of k can be set by experts on the database at the very beginning or by experiments. The determination of parameter k will be discussed in next subsection. *DPset* keeps the DPs or

EPs of the currently processed cluster. The objects in *DPset* are used to expand the corresponding cluster. Once a DP or EP's kNB is moved into the current cluster, it is removed from *DPset*. A cluster is completely detected when there is no object in *DPset*. When the NBC algorithm comes to stop, the unclassified objects whose *clst_no* property is NULL are regarded as noises or outliers.

The NBC algorithm starts with the *CalcNDF* function to calculate kNB, R-kNB and NDF of each object in *Dataset*. Among the traditional index structures, R^*-Tree and X-tree are usually used to improve the efficiency of kNB query processing over relatively low dimensional datasets. However, there is few index structure works efficiently over high-dimensional datasets. To tackle this problem, we employ a cell-based approach to support for kNB query processing. The data space is cut into high-dimensional cells, and VA file [3] is used to organize the cells. Due to space limitation, we neglect the detail here.

```
NBC(Dataset, k) {
   for each object p in Dataset
     p.clst_no=NULL; // initialize cluster number for each object

   CalcNDF(Dataset, k); // calculate NDF
   NoiseSet.empty(); // initialize the set for storing noise
   Cluster_count = 0; // set the first cluster number to 0
   for each object p in Dataset{ // scan dataset
     if(p.clst_no!=NULL or p.ndf < 1) continue;
     p.clst_no = cluster_count; // label a new cluster
     DPSet.empty(); // initialize DPSet

     for each object q in kNB(p){
       q.clst_no = cluster_count;
       if(q.ndf>=1) DPset.add(q)}

     while (DPset is not empty){ // expanding the cluster
       p = DPset.getFirstObject();
       for each object q in kNB(p){
         if(q.clst_no!=NULL)continue;
         q.clst_no = cluster_count;
         if(q.ndf>=1) DPset.add(q);}
       DPset.remove(p);
     }
     cluster_count++;
   }

   for each object p in Dataset{ // label noise
     if(p.clst_no=NULL) NoiseSet.add(p);}
}
```

Fig. 2. The NBC algorithm in C pseudo-code

2.3 Algorithm Analysis

The Determination of k Value. The parameter k roughly determines the size of the minimal cluster in a database. According to the neighborhood-based notion of cluster and the process of the NBC algorithm, to find a cluster, we must first find at least one DP or EP whose R-kNB is larger than or equal to its kNB(*i.e.*, the value of NDF not less than 1). Suppose C is the minimal cluster w.r.t. k in database D, and p is the first DP or EP found to expand cluster C. All objects in kNB(p) are naturally assigned to C. Considering p itself, therefore the minimal size of C is k+1. So we can use the parameter k to limit the size of the minimal cluster to be found.

A cluster is a set of data objects that show some similar and unique pattern. If the size of a cluster is too small, its pattern is not easy to demonstrate. In such a case, the data behaves more like outliers. In experiments, we usually set k to 10, with which we can find most meaningful clusters in the databases.

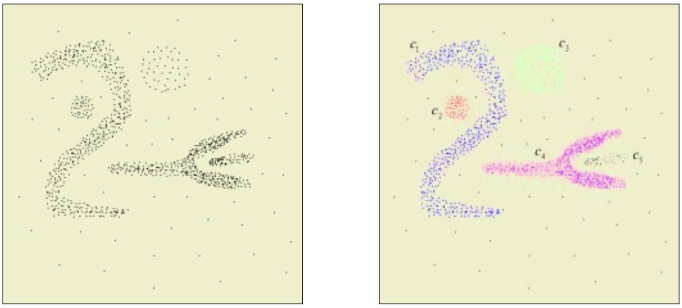
Complexity. The procedure of the NBC algorithm can be separated into two independent parts: calculating NDF and discovering clusters. The most time-consuming work of calculating NDF is to evaluate kNB queries. Let N be the size of the d-dimension dataset D. Mapping objects into appropriate cells takes $O(N)$ time. For a properly settled value of the cell length l, in average, cells of 3 layers are needed to search and each cell contains k objects. Therefore, the time complex of evaluating kNB query is $O(mN)$ where $m = k * 5^d$. For large datasets, $m \ll N$, it turns to $O(N)$. However, considering that $m \gg 1$, so time complexity of CalcNDF is $O(mN)$. The recursive procedure of discovering cluster takes $O(N)$. Therefore, the time complexity of the NBC algorithm is $O(mN)$.

3 Performance Evaluation

In this section, we evaluate the performance of the NBC algorithm, and compare it with DBSCAN. In the experiments, we take k=10. Considering that k value mainly affect the minimal cluster to find, we do not give the clustering results for different k values.

To test NBC's capability of discovering clusters of arbitrary shape, we use a synthetic dataset that is roughly similar to the database 3 in [2], but more complicated. In our dataset, there are five clusters and the noise percentage is 5%. The original dataset and the clustering result of NBC are shown in Fig.3. As is shown, NBC discovered all clusters and recognized the noise points.

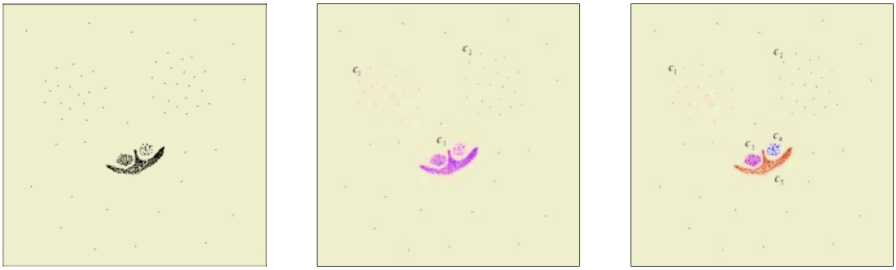
To demonstrate NBC's outstanding capability of discovering all clusters of different densities in one dataset, we use the synthetic dataset sample shown in Fig. 4(a). The clustering results by DBSCAN and NBC are shown in Fig. 4(b)(corresponding to a relatively low density threshold) and Fig. 4(c) respectively. We can see that NBC discovered all the five clusters. As for DBSCAN, no matter what density threshold we take, it can not detect all the five clusters. When the density threshold is selected low, DBSCAN can find the two sparse



(a) Original dataset

(b) Clustering result by NBC

Fig. 3. Discoverying clusters of arbitrary shape



(a) Dataset

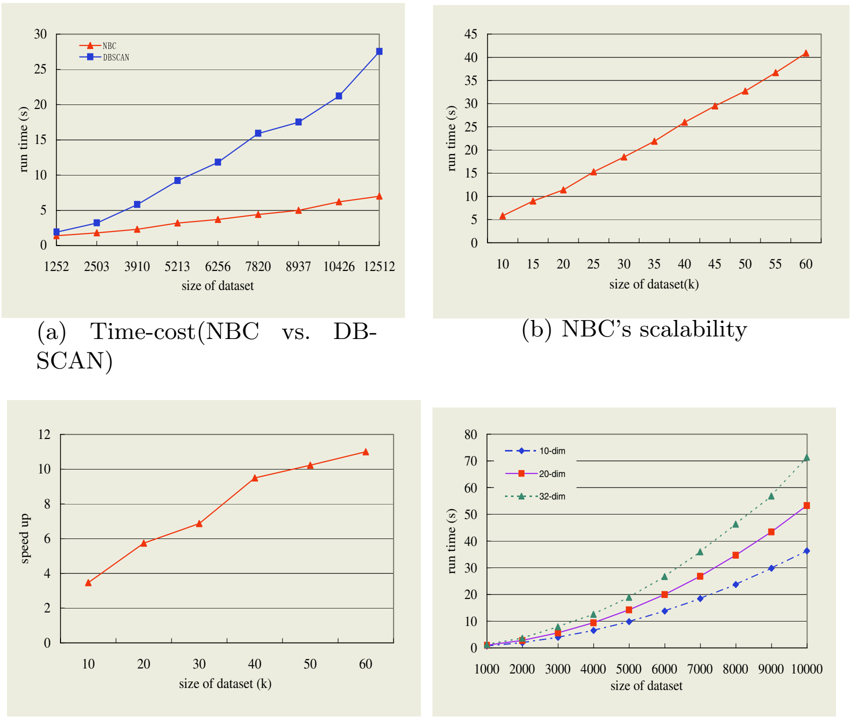
(b) DBSCAN's result

(c) NBC's result

Fig. 4. Discoverying clusters of different densities(NBC *vs.* DBCSAN)

clusters(C_1 and C_2), but the three dense clusters are merged into one cluster C_3(See Fig. 4(b)); While the threshold is set high, DBSCAN can find the three dense clusters(C_3, C_4 and C_5), but all data points in the two sparse clusters are labelled as noise(We do not illustrate the result here due to space limit).

To test the efficiency of NBC, we use the SEQUOIA 2000 benchmark databases [4] to compare NBC with DBSCAN. The time costs of NBC and DBSCAN over these datasets are shown in Fig. 5(a). As the size of dataset grows, time-cost of NBC increases slowly, while the time-cost of DBSCAN climbs quite fast. In Fig.5(b), we show the time cost of NBC on larger datasets (number of data points varies form 10,000 to 60,000). Here, we give only NBC's results because with the size of dataset increases, the discrepancy of time cost between DBSCAN and NBC is so large that their time costs cannot be shown properly in the drawing. In stead, we show the *speedup* of NBC over DBSCAN in Fig. 5(c). The time cost of NBC is linearly proportional to the size of dataset. And with the size of dataset increase, NBC has larger and larger speedup over DBSCAN.



(a) Time-cost(NBC vs. DBSCAN)

(b) NBC's scalability

(c) NBC's Speedup over DBSCAN

(d) High-dimensional data

Fig. 5. NBC clustering efficiency and scalability

To test the efficiency and scalability of NBC over high-dimensional datasets, we use the UCI Machine Learning Databases [5]. The results are in Fig.5(d), which show that the run-time of NBC on high-dimensional datasets is approximately linear with the size of database for middle high-dimensional data (d=10-20). But as the number of dimensions increases, the curve turns steeper. The reason is that higher dimensional data space will be divided into more cells, which means more neighbor cells will be searched for evaluating kNB queries.

4 Conclusion

We present a new clustering algorithm, NBC, *i.e.*, Neighborhood Based Clustering, which discovers clusters based on the neighborhood relationship among data. It can discover clusters of arbitrary shape and different densities. Experiments show that NBC outperforms DBSCAN in both clustering effectiveness and efficiency. More importantly, NBC needs fewer input parameter from the users than the existing methods.

References

1. J. Han and M. Kamber. Data mining: concepts and techniques. Morgan Kaufmann Publishers, 2000.
2. Ester M., Kriegel H., Sander J., and Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proc. KDD'96, pages 226-231, Portland, Oregon, 1996.
3. R. Weber, H.-J. Schek and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In Proc. of VLDB'98, pages 194-205, New York City, NY, August 1998.
4. M. Stonebraker, J. Frew, K. Gardels, and J. Meredith. The SEQUOIA 2000 Storage Benchmark. In Proc. of SIGMOD'93, pages 2-11, Washington D.C., 1993.
5. C. Merz, P. Murphy, and D. Aha. UCI Repository of Machine Learning Databases. At http://www.ics.uci.edu/ mlearn/MLRepository.html.