Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

A statistical information-based clustering approach in distance space

YUE Shi-hong (岳士弘)†[1], LI Ping (李 平)[1], GUO Ji-dong (郭继东)[2], ZHOU Shui-geng (周水庚)[1]

([1]*Institute of Industrial Process Control, Zhejiang University, Hangzhou 310027, China*)

([2]*Department of Mathematics, Yili Teacher's College, Yining 835000, China*)

†E-mail: shyue@iipc.zju.edu.cn

Received June 18, 2003; revision accepted Oct. 12, 2003

Abstract: Clustering, as a powerful data mining technique for discovering interesting data distributions and patterns in the underlying database, is used in many fields, such as statistical data analysis, pattern recognition, image processing, and other business applications. Density-based Spatial Clustering of Applications with Noise (DBSCAN) (Ester *et al*., 1996) is a good performance clustering method for dealing with spatial data although it leaves many problems to be solved. For example, DBSCAN requires a necessary user-specified threshold while its computation is extremely time-consuming by current method such as OPTICS, etc. (Ankerst *et al*., 1999), and the performance of DBSCAN under different norms has yet to be examined. In this paper, we first developed a method based on statistical information of distance space in database to determine the necessary threshold. Then our examination of the DBSCAN performance under different norms showed that there was determinable relation between them. Finally, we used two artificial databases to verify the effectiveness and efficiency of the proposed methods.

Key words: DBSCAN algorithm, Statistical information, Threshold
doi:10.1631/jzus.2005.A0071 **Document code:** A **CLC number:** TP391.41

INTRODUCTION

Clustering, groups database data into meaningful subclasses in such a way that minimizes the intra-differences and maximizes the inter-differences of these subclasses, is one of the most widely studied problems in data mining. Clustering technique is applied in many areas, such as statistical data analysis, pattern recognition, image processing, and other businesses applications. Up to now, many clustering algorithms have been proposed, in which famous algorithms contributed from the database community include classical k-NN (Han, 2001), DBSCAN (Ankerst *et al*., 1999), CURE (Guha *et al*., 1998), STING (Zhang *et al*., 1997), CLIGUE (Agrawal *et al*., 1998), WAVECLUSTER (Sheikholeslami *et al*., 1998), CHAMETEON (Karypos *et al*., 1993) and MSE (Nakamura and Kehtarnavaz, 1998). All these algorithms attempt to solve the clustering problems.

To apply the DBSCAN algorithm to large-scale spatial databases, this work aims

(1) to develop a method to cope with the open problem in DBSCAN algorithm, i.e. a necessary density threshold, which still fails to be solved efficiently now;

(2) to examine the performance of DBSCAN under different norms and obtain new insights;

(3) and based on the above two algorithms, to carry out three experiments to demonstrate their efficiency of handling outliers and verify their effectiveness.

This paper is organized as follows. Section 2 presents a summary on the DBSCAN algorithm, and analyzes its limitations and drawbacks when dealing

* Project (No. 2002AA412010-12) supported by the Hi-Tech Research and Development Program (863) of China

with large-scale databases. A method for determining the necessary density threshold is developed in Section 3. Two applications of artificial databases to demonstrate these algorithms' efficiency and effectiveness are given in Section 4. Section 5 presents the conclusion.

RELATED WORKS

DBSCAN algorithm

DBSCAN is a clustering algorithm that relies on a density-based notion of clusters. It is designed to discover the arbitrary-shaped clusters while being able to handle noise or outliers effectively. The key idea in DBSCAN is that for each data object of any cluster, the neighborhood of a given radius (EPS) has to contain at least a minimum number (Minpts) of objects. We give an overview of the major notions related to DBSCAN algorithm as follows (Ester *et al*., 1996).

Definition 1 (directly density-reachable) An object p is directly density-reachable from an object q with respect to (wrt) EPS and Minpts in the set of objects D if

(1) $p \in N_{EPS}(q)$ ($N_{EPS}(q)$ is the subset of D contained in the EPS-neighborhood of q).

(2) $Card(N_{EPS}(q)) \geq Minpts$, where $Card(\cdot)$ means the number of objects in a set.

Definition 2 (core object and border object) An object is a core object if it satisfies Condition 2 of Definition 1, and a border object is such an object that is not a core one itself but directly density-reachable from another core object.

Definition 3 (density-reachable) An object p is density-reachable from an object q wrt EPS and Minpts in the set of objects D, denoted as $p > \tilde{D} q$, if there exists a chain of objects $p_1, \ldots, p_n$, $p_1=q$, $p_n=p$ such that $p_i \in D$ and p_{i+1} is directly density-reachable from p_i wrt EPS and Minpts.

Definition 4 (density-connected) An object p is density-connected to an object q with respect to EPS and Minpts in the set of objects D if there exists an object $o \in D$ such that both p and q are density-reachable from o wrt EPS and Minpts in D.

Definition 5 (cluster) A cluster C wrt EPS and Minpts in D is a non-empty subset of D satisfying the following conditions:

(1) maximality: $\forall p, q \in D$, if $p \in C$ and $q > \tilde{D} p$ with respect to EPS and Minpts, then also $q \in C$.

(2) connectivity: $\forall p, q \in C$, p is density-connected to q with respect to EPS and Minpts in D.

Definition 6 (noise) Let $C_1, \ldots, C_k$ be the clusters with respect to EPS and Minpts in D, then we define the noise as the set of objects in D not belonging to any cluster C_i, i.e. noise=$\{p \in D | \forall i: p \notin C_i\}$.

The procedure for finding a cluster is based on the fact that a cluster can be determined uniquely by any of its core objects:

(1) Given an arbitrary object p for which the core object condition holds, the set $\{o \mid o > \tilde{D} p\}$ of all objects o density-reachable from p in D forms a complete cluster C and $p \in C$.

(2) Given a cluster C and an arbitrary core object $p \in C$, C equals the set $\{o \mid o > \tilde{D} p\}$.

To find a cluster, DBSCAN starts with an arbitrary object p in D and retrieves all objects of D density-reachable from p with respect to EPS and Minpts. If p is a border object, no objects are density-reachable from p and p is assigned to noise temporarily. Then DBSCAN handles the next object in database D. Retrieval of density-reachable objects is performed by successive region queries. A region query returns all objects intersecting a specified query region efficiently by R*-trees. Before clustering the database, R*-tree should be built in advance (Bechmann *et al*., 1990).

However, there are some DBSCAN algorithm problems limiting its applications. Here the most fundamental are the following well-known open problems:

P1 DBSCAN requires the user to specify a global threshold EPS (Minpts is often fixed to 4 to reduce the computational amount). In order to determine EPS, DBSCAN has to calculate the distance between an object and its kth (k=4) nearest neighbor for all objects. It sorts all objects according to the previously calculated distance and plots the sorted k-dist graph from OPTICS (Ankerst *et al*., 1999). In addition, DBSCAN is based on R*-tree or other analogous data structures, and calculates the k-dist value on the entire database. The two procedures are the most time-consuming phases in the whole clustering process, but their computational loads are not in-

cluded in time consumption as in $O(n\log n)$, so the actual time consumption of DBSCAN may be larger than that of $O(n\log n)$. Clustering procedure is very expensive so that it is computationally prohibitive for large databases. EPS and Minpts determine a density threshold, thus DBSCAN becomes a typical density-based clustering method. Furthermore, the Minpts usually is fixed to 4, thus the density threshold is perfectly determined by EPS.

P2 In most cases, the performance of an existing clustering algorithm is different under different norms. We have examined the performance of DBSCAN and obtained some new insights.

P3 In fact, a single threshold can hardly distinguish all clusters in a large spatial database when there are density-skewed clusters. In order to cope with similar problems, there exist algorithms such as CHAMELETON, etc. adopting different thresholds by partitioning the database. Database partitioning is favorable for clustering efficiency, but the partitioning technique is not easy to implement. After implementing partition for a database, merging and synthesizing, partitioned sub-clusters with different EPS values should be adopted. This may lead to further uncertainty of clustering results when we fail to find a clear guidance rule. Furthermore, there appear problems on how and when one should partition the database.

In this paper we only focus on P1 and P2, while P3 and other problems will be discussed in Yue *et al*.(2004).

Clustering validity and popular norms

As said before, although DBSCAN algorithm can perform the clustering procedure in a database with EPS, it still leaves the user with the responsibility of selecting density thresholds leading to the discovery of acceptable clusters. This is, of course, a problem common with many other clustering algorithms. Such threshold settings usually require empirical methods to search for and are difficult to be determined, especially when there is no prior knowledge of them in real world.

The problem of how EPS is chosen is closely related to what number of clusters is chosen. Once all densities of clusters in a database are nearly consistent, the determination of number of clusters will be nearly the same for EPS. Motivated by that, we shall recall some results on the clustering validity. It is well known that the number of clusters is the most important parameter in the clustering results and determines the structure of the clustering space. In contrast to the number of clusters, other parameters have second order effects on the clustering results in the dataset. In present methods for testing cluster validity (Halkidi *et al*., 2002), it is very difficult to evaluate the actual number of clusters in a dataset especially when there are arbitrary-shaped clusters inside it even though the dataset belongs to low dimensional space. Nevertheless, we assume the largest density of object in each cluster is approximately consistent and discuss the difficulty of distinguishing density-skewed clusters in Yue *et al*.(2004). Now as far as DBSCAN is concerned, it is clear that the clustering outcome such as the number of clusters, prototype locations, and "belongingness" of samples are governed by EPS selection. Changing EPS results in the migration of prototypes as well as their creation and elimination. The fundamental issue addressed here is the determination of an optimal scale size of EPS in a fast and efficient way.

In order to supply a benchmark to compare the current validity, we recall the notion of lifetime and drift speed of the MSE algorithm (Nakamura and Kehtarnavaz, 1998), which reflects respectively the long lasting or persistent clustering and stability of prototypes or reaching a stable state of clusters. Both notions are closely related to scale size which is a generally the ease EPS. The survival duration of a scale-space blob over a range of scales is referred to as lifetime. Adopting the same terminology here, the term lifetime in MSC algorithm is defined as follows.

Definition 7 (lifetime τ)

$$\tau=\delta_{\max}(c)-\delta_{\min}(c) \tag{1}$$

where $\delta_{\max}(c)$ and $\delta_{\min}(c)$ denote the maximum and minimum scale sizes, respectively, with the number of clusters c.

When τ attains its maximal value, the corresponding number of clusters has its optimal value.

Definition 8 (drift speed ρ)

$$\rho(\delta):=\frac{1}{c(\delta)}\sum_{i=1}^{c(\delta)}\sqrt{\sum_{p=1}^{r}\left(\frac{\partial v_{ip}}{\partial\delta}\right)^2} \tag{2}$$

where v_{ip} indicates the location of the ith prototype in the pth dimension.

As can be noticed in the above definition, the drift speed changes as a result of varying the scale size. Drift speed gives an indication of the stability of prototypes. Prototypes are said to reach a stable state when their drift speed is minimized. It is therefore meaningful to define the locations of prototypes as the smallest value of the drift speed. This provides a way for obtaining an optimal scale size δ^* as

$$\delta^* := \min_{\delta}{}^{-1} \rho(\delta) \quad \forall \delta \in [\delta_{\min}(c^*), \delta_{\max}(c^*)] \tag{3}$$

For the special example, DBSCAN is clearly governed by the above results, which can be incorporated into many clustering algorithms.

What follows is threshold EPS design detailedly discussed in Section 3 below.

DETERMINING EPS FROM STATISTICAL INFORMATION

In this section, we utilize statistical information to acquire the density threshold by a fast and efficient way.

Main properties of DBSCAN algorithm

After examining the clustering procedure by DBSCAN, we can obtain some conclusions.

Proposition 1 Given a dataset, there exist the following results:

1) All core objects with respect to a higher density threshold or a lower EPS are completely contained in the set of core objects with respect to a lower density threshold.

2) A cluster disappears as EPS increases if and only if its core object of largest density disappears.

3) If two sets of aggregating objects belong to the same cluster, there exists at least a core object at their intersection. Furthermore, an object Q is an outlier of EPS when $Card(N_{EPS}(Q))=1$; an object Q is border object iff $2 \le Card(N_{EPS}(Q)) < 4$ and there exists a core object in $N_{EPS}(Q)$.

Results 1) to 3) in Proposition 1 can be deduced directly by the procedures of DBSCAN algorithm, and they supply guideline for our experiments and analysis. We notice that in order to find an optimal threshold similar to EPS, the searching procedure in CHAMELEON algorithm, etc. should be scanned many times in the database. However, if such a generalized programming for DBSCAN algorithm is employed, many more times of scanning are required than those in previous case; so that the DBSCAN algorithm scarcely has any advantages in time cost. Thus, we must perform special searching procedure for EPS.

Determination of EPS from statistical analysis

Some signs are firstly defined as follows. Set D consisting of n objects, there exist in total C_n^2 different distances between two arbitrary objects in D and let the set of all these distances be S. Following this, all distances in S may be decomposed into three classes: one consisting of intradistances (i.e., the within-cluster distances) and denoted by S_1; another consisting of interdistances (i.e., the between-cluster distances) and denoted by S_2, the last consisting of the distances to outliers in D and denoted by S_3. Clearly, $S_1 \cup S_2 \subset S$. Now, we expect to detect approximately the presence position of optimal EPS with the aid of analysis of distance space S on D. Our deduction begins with an interesting question, how many intradistances are contained in S? Let our analysis below answer it. Set D that consists of c clusters with the number of points as $n_1, n_2, \ldots, n_c$ in decreasing order; then the ratio of the number of intradistances to that of interdistances in S is

$$\sum_{k=1}^{c} C_{n_k}^2 : \sum_{m=1}^{c-1} \sum_{k=m+1}^{c} C_{n_m}^1 \cdot C_{n_k}^1 \tag{4}$$

A direct calculation of Eq.(4) rarely gives us the generalized conclusion as there may be countless combinations of $n_1, n_2, \ldots, n_c$, so we resort to using an arithmetic series to approximate this number group of clusters. Let S be an arithmetic series such as a, $a-d$, $a-2d$, …, $a-(c-1)d$ with tolerance d approximating n_1, n_2, …, n_c, where a is the largest number of this series and $n=c[2a+(c-1)d]/2$. Then as $a \to \infty$, the limitation on the number of intradistances versus that of total distances in S, which is the modified version of Eq.(4), can be represented as

$$\lim_{a \to \infty} \sum_{m=1}^{c} C_{a-(m-1)d}^2 / C_n^2 = 1/(4c) \tag{5}$$

It follows that

$$\begin{aligned}&\lim_{a\to\infty}\sum_{m=1}^{c}C_{a-(m-1)d}^2/C_n^2\\&=\lim_{a\to\infty}\left\{\frac{a(a-1)}{2}+\frac{(a-d)(a-d-1)}{2}+\ldots\right.\\&\quad\left.+\frac{(a-(c-1)d)(a-(c-1)d-1)}{2}\right\}\Big/\frac{n(n-1)}{2}\\&=\lim_{a\to\infty}\{a(a-1)+\ldots+(a-(c-1)d)(a-(c-1)d-1)\}\\&\quad\div\{c(2a+(c-1)d)[c(2a+(c-1)d)-1]/4\}\end{aligned}$$

This result reveals an important fact that relative to the total number of distances in S, the number of intradistances is by far less than that of interdistances when D is a large spatial dataset. Of course, Eq.(5) holds only if the distance number of each cluster tends to consistence or $d\to 0$. However, there exist differences among all numbers of clusters in general. In order that these different cases can be resolved, we set $d=\delta a$, $0<\delta\leq 1$, obtaining

$$\lim_{a\to\infty}\sum_{m=1}^{c}C_{a-(m-1)d}^2/C_n^2=[1+(c-1)\delta^2)]/4c^2\approx\delta^2/(4c) \tag{6}$$

For example, in a large dataset with 20 clusters and δ=0.2, we approximately obtain results by Eq.(6) as 0.01. Now we rearrange all distances in S in increasing order. Most intradistances are smaller than interdistances and have closer correlation to EPS than the interdistances have. For any point in D, there exist equally C_{n-1}^2 distances from it to the rest of the points. If we set a threshold ε between maximal distance $d_{\max}$ and minimal $d_{\min}$, then S is partitioned into two sets, S_1 with smaller distances and S_2 with larger ones. We can decide that the larger densities of points would gain the larger presence frequency in S_1; and that most members in S_1 are intradistances in general. If we let the threshold be EPS in DBSCAN, then when the presence times of a point are higher than 4, it should be a core point, but not a border point or an outlier. Thus, determination of an accurate threshold is the key to determination of core points. After all, the intradistances comprise a trivial part of members in S. It is clear that they have numerical advantages in S_1 over distances of the same scale, because they are located in the dense areas. We have a threshold EPS partition S into two components, S_1 and S_2 and let ε_0 refer to the top $[\delta^2/(4c)C_n^2]$ minimal distances in S_1. After ranking all points in D, we retrieve points one by one according to the presence frequency in S_1 till $[\delta^2/(4c)C_n^2]$. How can we determine the values of the parameters δ and c in Eq.(6)? We shall determine c by the validity index in Eq.(1) and δ by drift speed in Section 4. In addition, c is the experimental parameter, and there exist many comparable results such as the equation $c\leq\sqrt{n}$ (Halkidi *et al*., 2002). Taking the number of clusters as c in Eq.(6) is a good strategy. We can obtain a fast and efficient equation for EPS whose searching procedure is given below. The remaining step is devoted to the clustering procedure like that of DBSCAN, whose results are the number of clusters by validity index and EPS, shown in Table 1.

Table 1 Pseudo code of DBSI algorithm

Determination of EPS based on statistical information (DBSI)
Construct distance space S.
Calculate the number of clusters, c, by Eqs.(1) and (2);
Construct intradistance space S_1.
Calculate δ by Eq.(3);
Search the top $[\delta/(4c^2)C_n^2]$ distances in S.
Return real number of clusters by validity index and EPS.

ACTIONS OF DBSCAN UNDER DIFFERENT NORMS

Most of clustering algorithms are very parameter-sensitive, i.e., a slight change of settings may lead to considerably different clustering partition of objects in a database, thus weakening its stabilization. Simultaneously, notice that to minimize intradistance and to maximize interdistance are necessary clustering equipments. We construct the validity index in terms of Eq.(1) and examine the clustering stabilization by different norms and Eq.(2). DBSCAN was implemented via the following steps:

1. Compute lifetime s by using Eq.(1);

2. Identify appropriate number of clusters c^* by using Eq.(2);

3. Compute the appropriate δ by the drift speed

in Eq.(3).

The first and second steps provide the number of clusters. Clearly if the number of clusters is known to begin with, these steps can be bypassed. The third step generates the difference values of the arithmetic series. In order to calculate Eq.(3), the appropriate objective function based on δ should be built up. In this paper, we employ the total sum of the interdistances between two arbitrary spherical neighborhoods of the prototype through their total sum of least squares. The behavior of typical DBSCAN with different norms has not been analyzed by applicable theoretical results. Now we first present new insight in order to provide theoretical support for explanations of our experiments.

When a small number of samples are available for clustering, the results under different norms give a rather large distinction. The essential reason lies, we think, in the difference of geometric distance on different norms. We try to make a quick comparison between various types of L_p norms that are the most popular norms and considered in this paper. Let us consider two arbitrary points a, b, for the sake of simplicity, in a 2-dimensional Euclidean space in Fig.1.

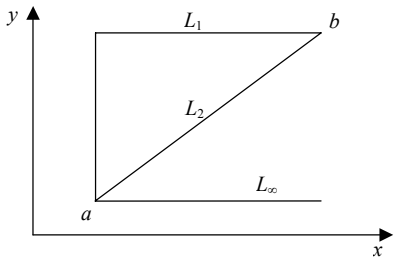

Fig.1 Three different norms of L_p

We can think of the L_2 norm distance as the distance from one point to the other in the (arbitrary) slope defined by those points. With the L_1 norm, we can measure the distance considering that we could move only horizontally or vertically from one point to another, and summing both the horizontal and vertical movements. With the L_∞ norm, again we can move only horizontally or vertically from one point to another, but only the biggest of the horizontal or vertical distances matters. We can easily see that the "paths" for measuring each of the distances form a right-angled triangle; the L_2 norm distance is the size of the hypotenuse, the L_1 norm distance is the sum of the sizes of cathetus, and the L_∞ norm distance is the largest cathetus. Clearly, for any points a, b the inequality between the distances in the three different norms:

$$\| a-b \|_\infty \leq \| a-b \|_2 \leq \| a-b \|_1 \tag{7}$$

holds. Therefore, we can think of L_2 norm as something between L_∞ norm and L_1 norm. Note that for points in a line parallel to one of the axes, Eq.(7) attains the equal sign. It is obvious that the varying rates under different norms are different for the same series of possible distances. As DBSCAN is concerned with a small amount of samples, we can expect that when point b is taken as the prototype of a cluster, the larger distance values between two points or the higher resolution rate are available. Intuitively, this case is just like looking at a cluster with different diploid magnifiers or clustering norms. Unfortunately, as the scale of different norms increases, this noise will have serious effects since its action is magnified. Another observation that we can make for DBSCAN in this paper is that as the number of test samples gets larger, and the clustering results distinctly as different norms get narrower when the test samples are extremely dense. Consequently, the results of using L_1, L_2 and L_∞ norms will be closer and closer to each other and applicable samples cannot be dense enough, so the analysis of distinction under different norms is necessary.

We have observed that good clustering performance is closely related to the magnifier for border objects. An extremely important result is that if we choose the average density of all border objects as the benchmark, then the largest magnifier time by different norms may gain the best clustering results. Some explainable examples are exhibited in experiments in Section 5. We think these results are inevitable, because the case corresponds to the case when the boundary of clusters is the clearest same time, they are most stabilized.

EXPERIMENTS

Actions of DBSCAN under different norms

The experiment was performed in real dataset in

2-dimensional space. The sample dataset contained total 2000 samples. There were 20 clusters in which six clusters contained less than 20 samples; the largest one contained 1220 samples while the smallest one contained only 10 samples. In addition, they were density-skewed. We use DBSCAN under different norms. The clustering results are listed in Table 2. The results showed that if the average density of all border points was close to the density determined by EPS in DBSCAN, the accuracy was the highest; on the contrary that larger difference in density led to lower accuracy. The stabilization of clusters was also examined in order to verify the relation between the stabilization index and the different norms. When the average density of border objects gained the largest magnification by a norm, the stabilization was the best.

Table 2 Results of DBSCAN under different norms

Norms	L_1	L_2	L_3	L_6	L_∞
Average density	1.1	1.0	1.4	1.3	0.9
Missing samples	18/4.1	24/3.2	10/10.2	13/1.2	29/1.3
Missing clusters	9	8	3	3	5
Run time (s)	14	17	15	7	25

Notes: *EPS*=12.00, where "/" corresponds to the results under different norms

Comparison of the clustering results based on DBSI and *k*-dist graph

This test was carried out to compare the efficiency and effectiveness on the DBSI and *k*-dist for determining EPS. The source dataset came from the database SIMEIM (http://home.wi.rr.com/wiscwx/). Most clusters had normal distribution in 8-dimensional space; and the test was performed for clustering associated with six groups of data including five clusters with 100 samples, 11 clusters with 500 samples, 26 clusters with 2000 samples, 35 clusters with 4000 samples, 45 clusters with 5000 samples and 60 clusters with 6000 samples. These corresponding clustering results are shown in Fig.2 and Fig.3. The first and second datasets overlapped and there were different number of clusters in which the maximal cluster contained 40 and the minimal cluster contained only 19 samples.

The third and fourth datasets contained density-skewed clusters, all of which had similar densities. The fifth and sixth datasets included two of the above cases simultaneously. In time cost for the same

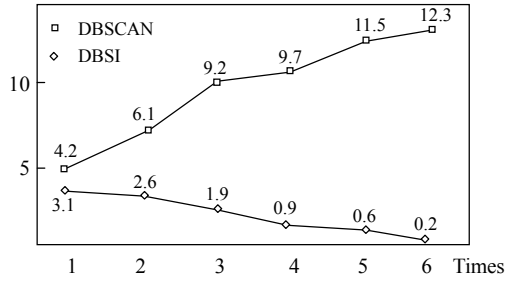

Fig.2 Comparison of run time

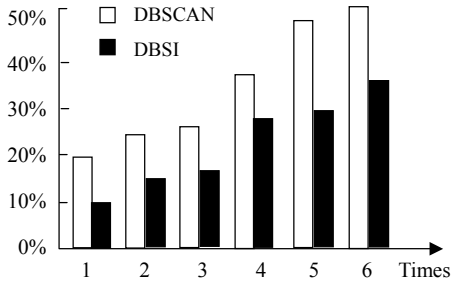

Fig.3 Comparison of missing ratio

dataset, the clustering algorithm based on *k*-dist graph requires much more than that in DBSI; which is the reason why we give more emphasis on the latter, but the former gives higher accuracy that was reflected by the missing ratio of samples. The larger the number of clusters, the larger was the difference. A high accuracy over a wider range than that of the former, but whether this result is inevitable is still unsolved and will be further discussed.

CONCLUSION

In this paper, although our discussion has largely focused on 2-dimensional space, application of the proposed algorithms to higher dimensional database (e.g., 2–10 dimensional) should be no problem. With the DBSCAN algorithm in this paper, we introduce an approach based on statistical information instead of OPTICS by which EPS can be searched in wider range. We evaluated comprehensively the performance of the new algorithms through using three experiments, which showed that these algorithms are effective and efficient in clustering large spatial databases; and that DBSI has almost the same accuracy as OPTICS. After considering the choice of different

norms, the results of DBSI seem to be better. However, some assumptions in the paper are not valid for arbitrary circumstances such as the series-tolerance-based approach. Establishing an ideal adaptive and interactive density-based clustering algorithm that needs as little user involvement as possible is our aim in this work.

References

Agrawal, R., Gehrke, J., Gunpopulos, D., Raghavan, P., 1998. Automatic Subspace Clustering of High DiMensional Data for Data Mining Applications. Proc. of ACM SIGMOD Int. Conf. on Management of Data, Seattle, WA, p.73-84.

Ankerst, M., Breunig, M., Kriegel, H.P., Sander, J., 1999. OPTICS: Ordering Points to Identify the Clustering Structure. Proc. 1999 ACM SIGMOD Int. Conf. Management of Data Mining, PA, p.49-60.

Bechmann, N., Kriegel, H.P., Schneider, R., Seeger, B., 1990. The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. Proc. ACM SIGMOD Int. Conf. On Management of Data. Alt. City, NJ, p.322-331.

Ester, M., Kriegel, H.P., Sander, H., XU, X., 1996. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Proc. of 2nd Int. Conf. on Knowledge Discovering in Databases and Data Mining. Portland, Oregon, p.232-1239.

Guha, S., Rastogi, R., Shim, K., 1998. CURE: An Efficient Clustering Algorithm for Large Databases. Proc. of the ACM SIGMOD Int. Conf. on Management of Data. Seattle, WA, p.73-84.

Han, J., 2001. Data Mining. Morgan Kaufmann Publishers, USA, p.242-266.

Halkidi, M., Batistakis, Y., Vazirgiannis, M., 2002. Clustering validity checking methods: part II. *SIGMOD Record*, **31**(4):51-62.

Karypos, G., Han, E.H., Kunar, V., 1993. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *Computer*, **32**(8):68-75.

Nakamura, E., Kehtarnavaz, N., 1998. Determining number of clusters and prototype locations via multi-scale clustering. *Pattern Recognition Letters*, **19**(3):1265-1283.

Sheikholeslami, G., Chatterjee, S., Zhang, A., 1998. Wavecluster: A Multi-resolution Clustering Approach for very Large Spatial Databases. Proc. of 24th VLDB Conf., New York, p.428-439.

Yue, S.H., Li, P., Guo, J.D., Zhou, S.G., 2004. Using Greedy algorithm: DBSCAN revisited II. *J Zhejiang Univ SCI*, **5**(11):1405-1412.

Zhang, W., Yang, Y., Munta, R., 1997. STING: An Statistical Information Grid Approach to Spatial Data Mining. Proc. of 23rd VLDB Conf., Seattle, WA, p.186-195.