Available online at www.sciencedirect.com
ScienceDirect
Information Systems 32 (2007) 978–986
www.elsevier.com/locate/infosys

A local-density based spatial clustering algorithm with noise

Lian Duan[a,*], Lida Xu[b,d], Feng Guo[c], Jun Lee[a], Baopin Yan[a]

[a]*Computer Network Information Center, Chinese Academy of Sciences, Beijing, China*
[b]*The Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China*
[c]*Zhejiang University, Hangzhou, China*
[d]*Old Dominion University, VA, USA*

Abstract

Density-based clustering algorithms are attractive for the task of class identification in spatial database. However, in many cases, very different local-density clusters exist in different regions of data space, therefore, DBSCAN method [M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: E. Simoudis, J. Han, U.M. Fayyad (Eds.), Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, AAAI, Menlo Park, CA, 1996, pp. 226–231] using a global density parameter is not suitable. Although OPTICS [M. Ankerst, M.M. Breunig, H.-P. Kriegel, J. Sander, OPTICS: ordering points to identify the clustering structure, in: A. Delis, C. Faloutsos, S. Ghandeharizadeh (Eds.), Proceedings of ACM SIGMOD International Conference on Management of Data Philadelphia, PA, ACM, New York, 1999, pp. 49–60] provides an augmented ordering of the database to represent its density-based clustering structure, it only generates the clusters with local-density exceeds certain thresholds but not the cluster of similar local-density; in addition, it does not produce clusters of a data set explicitly. Furthermore, the parameters required by almost all the major clustering algorithms are hard to determine although they significantly impact on the clustering result. In this paper, a new clustering algorithm LDBSCAN relying on a local-density-based notion of clusters is proposed. In this technique, the selection of appropriate parameters is not difficult; it also takes the advantage of the LOF [M.M. Breunig, H.-P. Kriegel, R.T. Ng, J. Sander, LOF: identifying density-based local outliers, in: W. Chen, J.F. Naughton, P.A. Bernstein (Eds.), Proceedings of ACM SIGMOD International Conference on Management of Data, Dalles, TX, ACM, New York, 2000, pp. 93–104] to detect the noises comparing with other density-based clustering algorithms. The proposed algorithm has potential applications in business intelligence.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Data mining; Local outlier factor; Local reachability density; Local-density-based clustering

1. Introduction

As larger amounts of data are collected and stored in databases, the need for efficiently and effectively analyzing and utilizing the information contained in the data has been increasing. One of the primary data mining techniques is cluster analysis which partitions a data set into groups so that the points in one group are similar to each other. In recent years, cluster analysis with improving algorithms has been the focus of a huge amount of research effort since although a large number of clustering algorithms have been developed but none of them is suitable for all types of applications.

*Corresponding author.
E-mail address: duanlian@cstnet.cn (L. Duan).

0306-4379/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.is.2006.10.006

The goal of a clustering algorithm is to decompose a data set into a set of meaningful groups. It seems that there is always room for a new clustering algorithm that is more efficient or better suited to an application. There are three related reasons why the effectiveness of clustering algorithms has been limited. First, the values for input parameters required by clustering algorithms are usually difficult to determine; it is especially true for those real world data with high dimensionality. Second, cluster algorithms are sensible to those parameter values, often producing very different partitions of the data set even for slightly different parameter settings. Third, real world data sets with high dimensionality often have a much skewed distribution that is difficult to be revealed by a clustering algorithm using only one global parameter setting.

In this paper, a new algorithm of local-density-based cluster analysis is proposed. In this new algorithm, the concept of *local outlier factor* (LOF) for each object in a data set is introduced which represents the degree of outlierness [1]. LOF and *local reachability density* (LRD) are used later to detect clusters in a data set and the noise data that do not belong to any of those clusters.

The paper is organized as follows. Related work on density-based clustering is briefly discussed in Section 2. In Section 3, the notion of cluster is presented which is based on the concept of local-density; the basic notions of LOF and LRD are also introduced. In Section 4, the algorithm called LDBSCAN which is intended to discover different local-density clusters in a spatial database is presented. Meanwhile, the impact of parameters and how to choose appropriate values for LDBSCAN is discussed. In Section 5, an experimental evaluation of the effectiveness and efficiency of LDBSCAN using synthetic data is presented. Section 6 concludes the paper with a summary and some directions for future research.

2. Related work

Density-based clustering has been very popular in data mining. The approach taken by the density-based clustering algorithms is to apply a local cluster criterion. Clusters are formed as regions in the data space in which the objects are of similar density, and which are separated by regions with different object density. These regions may have an arbitrary shape and the points inside a region may be arbitrarily distributed.

A common way to find regions of high-density in the data space is based on grid cell densities [2]. A histogram is constructed by partitioning the data space into a number of non-overlapping regions or cells. Cells containing a relatively large number of objects are potential cluster centers and the boundaries between clusters fall within the "valleys" of the histogram. The outcome of this method depends on the size of the cells which must be specified by the user. Cells of small volume will give a very "noisy" estimate of the density, whereas large cells tend to overly smooth the density estimate.

A density-based clustering method has been proposed by Ester et al. [3] which is not grid-based. The basic idea of the algorithm DBSCAN is that for each point of a cluster the neighborhood of a given radius (ε) has to contain at least a minimum number of points (MinPts) where ε and MinPts are input parameters.

Another algorithm OPTICS [4] creates an augmented ordering of the database representing its density-based clustering structure. This cluster-ordering contains information which is equivalent to the density-based clustering corresponding to a broad range of parameter settings. However, it only generates the clusters whose local-density exceeds some threshold instead of similar local-density clusters and does not produce clusters of a data set explicitly.

The density-based algorithm WaveCluster [5] applies wavelet transformation to the feature space [6,7]. The algorithm is able to detect clusters of arbitrary shape at different scales with complexity of $O(n)$. The algorithm is grid-based and only applicable to low-dimensional data. Input parameters include the number of grid cells for each dimension, the wavelet to use, and the number of applications of the wavelet transformation.

The density-based algorithm DENCLUE [8] uses a grid which is very efficient since it only keeps information about grid cells that do actually contain data points and manages these cells in a tree-based access structure. This algorithm generalizes some other clustering approaches which, however, resulting in a large number of input parameters.

The density- and grid-based clustering technique CLIQUE [9] has been proposed for data mining in high-dimensional data space. Input parameters are the size of the grid and a global density threshold for clusters. The major difference between this algorithm and all other clustering approaches is that this method also detects subspaces of the highest dimensionality as high-density clusters exist in those subspaces.

CURD [10] captures the shape and extent of a cluster by references, and then analyzes the data based on the references. It is able to discover clusters with arbitrary shape and is insensitive to noise data. Mining very large databases is its goal; however, the effectiveness is a problem.

A new shifting grid clustering algorithm [11] uses the concept of shifting grid. The algorithm is a non-parametric type, which does not require users inputting parameters. It divides each dimension of the data space into certain intervals to form a grid structure in the data space. Based on the concept of sliding window, shifting of the whole grid structure is introduced to obtain a more descriptive density profile. It clusters data in terms of cell rather than points.

3. Basic notions of LDBSCAN

3.1. Problems of existing approaches

An important property of many real world data sets is that their intrinsic cluster structures are unable to be characterized by global density parameters. As a result, very different local densities may be needed to reveal clusters in different regions of the data space. For example, in the data set depicted in Fig. 1, it is impossible to detect the clusters A, B, C_1, C_2, and C_3 simultaneously using one global density parameter. A global density-based decomposition would be needed for the clusters A, B, and C, or C_1, C_2, and C_3. In the second case, the objects from A and B may be noise only.

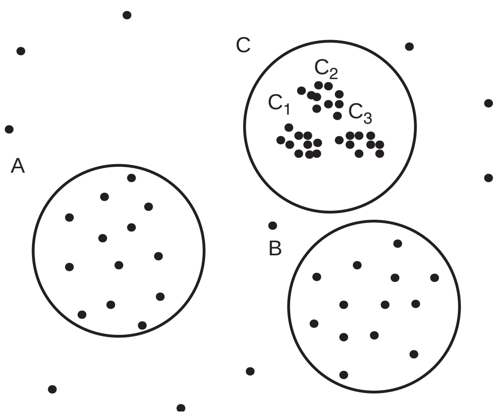

Fig. 1. Clusters with respect to different densities [4].

OPTICS can solve this problem; however, it only creates an augmented ordering of the database representing its density-based clustering structure instead of producing clusterings of a data set explicitly. In addition, it is short at generating the overlap clusters appropriately which is discussed in Section 5.1. Therefore, a local-density-based decomposition which consists of the clusters A, B, C_1, C_2, and C_3 explicitly is needed.

3.2. Formal definition of LRD and LOF

The LOF of each object represents the degree the object is being outlying and the LRD of each object represents the local-density of the object. The formal definitions for these notions of LOF and LRD are shortly introduced in the following. More details are provided in [1].

Definition 1 (*k-Distance of an object p*). For any positive integer k, the *k-distance of object p*, denoted as *k-distance*(p), is defined as the distance $d(p,o)$ between p and an object $o \in D$ such that:

(1) for at least k objects $o' \in D \backslash \{p\}$ it holds that $d(p,o') \leqslant d(p,o)$, and
(2) for at most $k-1$ objects $o' \in D \backslash \{p\}$ it holds that $d(p,o') < d(p,o)$.

Definition 2 (*k-Distance neighborhood of an object p*). Given the k-distance of p, the *k-distance neighborhood of p* contains every object whose distance from p is not greater than the k-distance, i.e. $N_{k\text{-}distance(p)}(p) = \{q \in D\backslash\{p\} | d(p,q) \leqslant k\text{-}distance\ (p)\}$. These objects q are called the k-nearest neighbors of p.

As no confusion arises, the notation can be simplified to use $N_k(p)$ as a shorthand for $N_{k\text{-}distance(p)}(p)$.

Definition 3 (*Reachability distance of an object p w.r.t. object o*). Let k be a natural number. The *reachability distance* of object p with respect to object o is defined as

$$reach\text{-}dist_k(p,o) = \max\{k\text{-}distance(o), d(p,o)\}.$$

Definition 4 (*LRD of an object p*). The LRD of p is defined as

$$LRD_{MinPts}(p) = 1 \Bigg/ \left(\frac{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p,o)}{|N_{MinPts}(p)|} \right).$$

Intuitively, the LRD of an object p is the inverse of the average *reachability distance* based on the *MinPts*-nearest neighbors of p.

Definition 5 (*LOF of an object p*). The LOF of p is defined as

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} LRD_{MinPts}(o)/LRD_{MinPts}(p)}{|N_{MinPts}(p)|}.$$

The LOF of object p is the average of the ratio of the LRD of p and those of p's *MinPts*-nearest neighbors. It captures the degree to which p is called an outlier. It is not difficult to see that the higher the ratio of the LRD of p and those of p's *MinPts*-nearest neighbors is, the farther away the point p is from its nearest cluster, and the higher the LOF value of p is.

Since the LOF represents the degree the object is being outlying and the LOF of most objects p in a cluster is approximately equal to 1, we regard object p must belong to certain cluster if $LOF(p) \leqslant LOFUB$.

3.3. A local-density based notion of clusters

When looking at the sample set of points depicted in Fig. 2, we can easily and unambiguously detect clusters of points and noise points not belonging to any of those clusters.

The main reason why we recognize the clusters is that within each cluster we have a typical local-density of points which is different from the outside of the cluster.

In the following, this intuitive notion of "clusters" and "noise" will be formalized. Note that both notion of clusters and algorithm LDBSCAN apply well to 2D or 3D Euclidean space as to some high-dimensional feature space. The key idea is that for any point p, $LOF(p) \leqslant LOFUB$, i.e. point p is not an outlier and belongs to certain cluster C, if point q is the *MinPts*-nearest neighbor of p and has the similar LRD with p, q belongs to the same cluster C of p. This approach works with any distance function so that an appropriate function can be chosen for a given application. In this paper, for the purpose of proper visualization, all examples will be in 2D space using the Euclidean distance.

Definition 6 (*Core point*). A point p is a *core point* w.r.t. $LOFUB$ if $LOF(p) \leqslant LOFUB$.

If $LOF(p)$ is small enough, it means point p is not an outlier and must belong to certain cluster. Therefore it can be regarded as a core point.

Definition 7 (*Directly local-density-reachable*). A point p is *directly local-density-reachable* from a point q w.r.t. *pct* and *MinPts* if

(1) $p \in N_{MinPts}(q)$, and
(2) $LRD(q)/(1+pct) < LRD(p) < LRD(q) * (1+pct)$.

Here, the parameter *pct* is used to control the fluctuation of local-density. However, in general, it is not symmetric if q is not the *MinPts*-nearest neighbor of p. Fig. 3 shows the asymmetric case. Let $MinPts = 3$ and $pct = 0.3$, we can calculate that $LRD(p)/LRD(q) = 1.2784$. It shows that p is directly local-density-reachable from q, but q is not directly local-density-reachable from p.

Definition 8 (*Local-density-reachable*). A point p is *local-density-reachable* from the point q w.r.t. *pct* and *MinPts* if there is a chain of points $p_1, p_2, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly local-density-reachable from p_i.

Local-density-reachability is a canonical extension of direct local-density-reachability. This relation is transitive, but it is not symmetric. Fig. 4 depicts the relations of some sample points and, in particular, an asymmetric case. Let $MinPts = 3$, $pct = 0.3$. According to the above definitions,

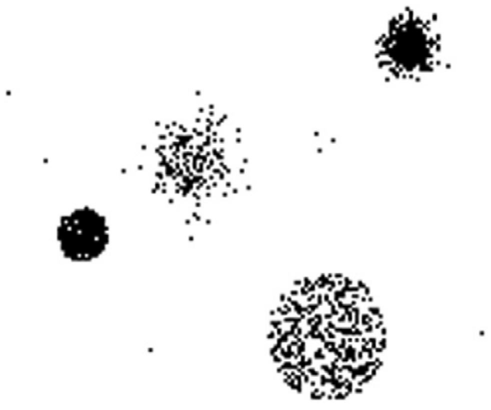

Fig. 2. Sample data [1].

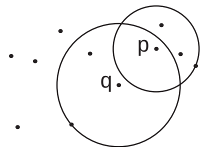

Fig. 3. Directly local-density-reachability.

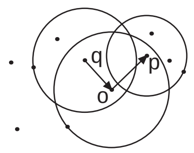

Fig. 4. Local-density-reachability.

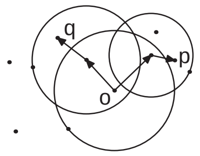

Fig. 5. Local-density-connectivity.

$LRD(p)/LRD(o) = 1.2784$, $LRD(o)/LRD(q) = 0.9538$. So we can see the asymmetric case between q and p.

Definition 9 (*Local-density-connected*). A point p is *local-density-connected* to a point q from o w.r.t. pct and $MinPts$ if there is a point o such that both p and q are local-density-reachable from o w.r.t. pct and $MinPts$.

From the definition, local-density-connectivity is a symmetric relation shown in Fig. 5.

Now we are able to define the notion of cluster that is local-density-based. Intuitively a cluster is defined as a set of local-density-connected points which is maximal w.r.t. local-density-reachability. Noises are defined relatively to a given set of clusters. Noises are simply the set of points in D not belonging to any of its clusters.

Definition 10 (*Cluster*). Let D be a database of points, and point o is a selected core point of C, i.e. $o \in C$ and $LOF(o) \leqslant LOFUB$. A *cluster* C w.r.t. $LOFUB$, pct, and $MinPts$ is a non-empty subset of D satisfying the following conditions:

(1) $\forall p : p$ is local-density-reachable from o w.r.t. pct and $MinPts$, then $p \in C$. (Maximality),
(2) $\forall p, q \in C : p$ is local-density-connected q by o w.r.t. $LOFUB$, pct and $MinPts$. (Connectivity).

Definition 11 (*Noise*). Let $C_1, \ldots, C_k$ be the clusters of the database D w.r.t. parameters $LOFUB$, pct, and $MinPts$. Then we define the *noise* as the set of points in the database D not belonging to any cluster C_i, i.e. $noise = \{p \in D \mid \forall i : p \notin C_i\}$.

Given the parameters $LOFUB$, pct, and $MinPts$, clusters can be found with a two-step approach. First, an arbitrary point p from the database satisfying the core point condition $LOF(p) \leqslant LOFUB$ as a seed is chosen. Second, all points that are local-density-reachable from the seed obtaining the cluster containing the seed are retrieved.

4. LDBSCAN: local-density-based spatial clustering of applications with noise

In this section, we present the algorithm LDBSCAN (local-density-based spatial clustering of applications with noise) which is designed to discover the clusters and the noise in a spatial database according to Definitions 10 and 11. First the appropriate parameters $LOFUB$, pct, and $MinPts$ of clusters and one core point of the respective cluster are selected. Then all points that are local-density-reachable from the given core point using the correct parameters are retrieved. Fortunately all the parameters are relative, and not absolute as those of DBSCAN. They are not difficult to be chosen and fall within certain range as presented in Section 4.2.

4.1. The algorithm

To find a cluster, LDBSCAN starts with an arbitrary point p and retrieves all points local-density-reachable from p w.r.t. $LOFUB$, pct, and $MinPts$. If p is a core point, this procedure yields a cluster w.r.t. $LOFUB$, pct, and $MinPts$ (see Definition 10). If p is not a core point, LDBSCAN will check the next point of the database.

In the following, we present a basic version of LDBSCAN without details of data types and generation of additional information about clusters:

```
LDBSCAN (SetOfPoints, LOFUB, pct, MinPts)
 // SetOfPoints is UNCLASSIFIED
 InitSet (SetOfPoints); // calculate LRD and LOF
 of eachpoint
 ClusterID := 0;
 FOR i FROM 1 TO SetOfPoints.size DO
   Point := SetOfPoints.get(i);
   IF Point.ClId = UNCLASSIFIED THEN
     IF LOF(Point) ⩽ LOFUB THEN // core
 point
       ClusterID := ClusterID + 1;
       ExpandCluster(SetOfPoints, Point,
 ClusterID, pct, MinPts);
     ELSE // no core point
```

```
            SetOfPoint.changeClId(Point,NOISE);
         END IF
      END IF
      END FOR
END; //LDBSCAN
```

SetOfPoints is the set of the whole database. *LOFUB*, *pct*, and *MinPts* are the appropriate parameters determined according to what is presented in Section 4.2. The function SetOfPoints.get(i) returns the ith element of SetOfPoints. Points which have been marked to be NOISE may be changed later if they are local-density-reachable from some core point of the database. The most important function used by LDBSCAN is ExpandCluster which is presented in the following:

```
ExpandCluster(SetOfPoints, Point, ClusterID, pct,
MinPts)
 SetOfPoint.changeClId(Point,ClusterID);
 FOR i FROM 1 TO MinPts DO
   currentP := Point.Neighbor(i);
   IF currentP.ClId IN {UNCLASSIFIED,NOISE}
        and DirectReachability(currentP,Point)
 THEN
      TempVector.add(currentP);
      SetOfPoint.changeClId(currentP,ClusterID);
   END IF
 END FOR
 WHILE TempVector <> Empty DO
   Point := TempVector.firstElement();
   TempVector.remove(Point);
   FOR i FROM 1 TO MinPts DO
      currentP := Point.Neighbor(i);
      IF currentP.ClId IN
 {UNCLASSIFIED,NOISE}
        and DirectReachability(currentP,Point)
 THEN
      TempVector.add(currentP);
      SetOfPoint.changeClId(currentP,ClusterID);
      END IF
   END FOR
 END WHILE
END; //ExpandCluster
```

The function DirectReachability(currentP,Point) is presented in the following:

```
DirectReachability(currentP,Point) : Boolean
  IF $LRD(currentP) > LRD(Point)/(1+pct)$
       and $LRD(currentP) < LRD(Point) * (1+pct)$
  THEN
     RETURN True;
  ELSE
     RETURN False;
END; //DirectReachability
```

4.2. Determining the parameters LOFUB, pct, and MinPts

In this section, we discuss how the result of LDBSCAN is influenced by the choice of the *LOFUB*, *pct*, and *MinPts*, and how the right values for LDBSCAN are determined.

There are two totally different parameters of *MinPts*. One is for the calculation of LOF and the other is for the clustering algorithm which will be discussed later in this section. For most of the data sets, it appears work well when *MinPts* for LOF picks 10–20 and more details can be found in [1]. For convenience of presentation, $MinPts_{LOF}$ is used as a shorthand of *MinPts* for LOF and $MinPts_{LDBSCAN}$ as a shorthand of *MinPts* for the clustering algorithm.

For objects deep inside a cluster, the LOFs are approximately equal to 1. The greater the LOF is, the higher possibility that for the object it is an outlier. More details about LOF are described in [1]. If the value that is selected for *LOFUB* is too small, some core points may be mistakenly considered as outliers; and if the value is too large, some outliers may be mistakenly considered as core points. For most of the data sets that have been experimented with, picking 1.5–2.5 appears to work well. However, it also depends. For example, we identified multiple clusters, e.g., a cluster of pictures from a tennis match and the reasonable *LOFUB* is up to 7. In Fig. 6, the red points are those whose LOF exceeds the *LOFUB* when $MinPtsLOF = 15$.

Parameter *pct* controls the local-density fluctuation as it is accepted. The value of *pct* depends on the fluctuation of the cluster. Generally speaking, it is restricted to [0.2, 0.5]. Of course in some particular situations, other values out of this range can be chosen. Let $MinPts_{LOF} = 15$, $MinPts_{LDBSCAN} = 10$, and $LOFUB = 2.0$. Fig. 7 shows the different clustering results with different values of *pct*.

Parameter $MinPts_{LDBSCAN}$ determines the stand-by objects belonging to the same cluster of the core point. Clearly $MinPts_{LDBSCAN}$ can be as small as 1. However, if $MinPts_{LDBSCAN}$ is too small, some reasonable objects may be missed. Thus the first

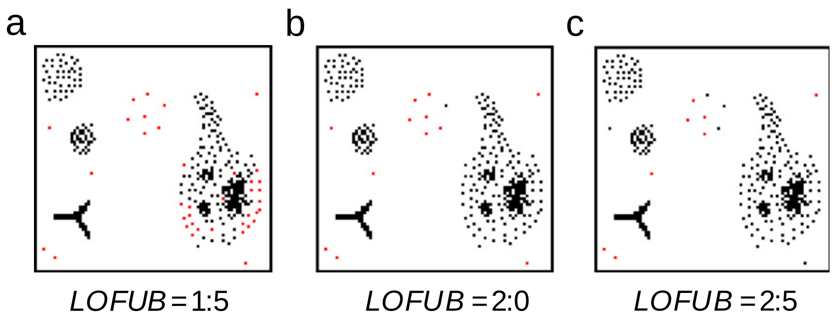

Fig. 6. Core points and outliers. (a) $LOFUB = 1.5$, (b) $LOFUB = 2.0$, (c) $LOFUB = 2.5$.

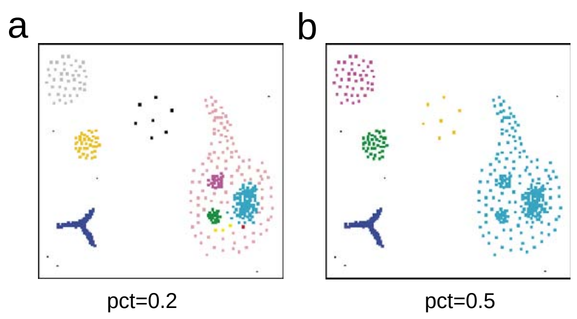

Fig. 7. Clustering results with different values of pct. (a) $pct = 0.2$, (b) $pct = 0.5$.

suggestion we provide for picking the LowerBound of $MinPts_{LDBSCAN}$ is that it should be at least 5 to take enough reasonable objects into account. The second suggestion we provide for the UpperBound of $MinPts_{LDBSCAN}$ is based on a more subtle observation.

Definition 12 (*Distance between the two clusters*). Let C_1, C_2 be the clusters of the database D. The distance between C_1 and C_2 is defined as

$$distance(C_1, C_2) = \min\{distance(p, q) \mid p \in C_1, q \in C_2\}.$$

Let $p \in C_1$, $q \in C_2$, C_1 has the similar density with C_2. p and q are the nearest objects between C_1 and C_2. Consider the simple situation that $distance(C_1, C_2)$ is small enough shown in Fig. 8, obviously that as $MinPts_{LDBSCAN}$ values increase, there will be a corresponding monotonic sequence of changes to *MinPts-distance(p)*. As the $MinPts_{LDBSCAN}$ values increase, once *MinPts-distance(p)* is greater than $distance(C_1, C_2)$, C_1 and C_2 will be generated into one cluster. In Fig. 8, clustering with any core point in C_1 is started. When $MinPts_{LDBSCAN}$ reaches 10, C_1 and C_2 will be

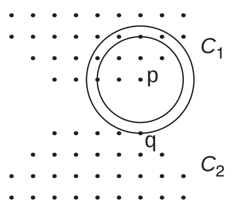

Fig. 8. Different values for $MinPts_{LDBSCAN}$.

generated into one cluster C. Therefore, the value for $MinPts_{LDBSCAN}$ should not be too large. When $MinPts_{LDBSCAN}$ reaches 15, enough candidates will be considered. The value ranges from 5 to 15 can be chosen for $MinPts_{LDBSCAN}$.

5. Experiments

In this section, we will demonstrate how the proposed LDBSCAN can successfully generate clusters which appear to be meaningful that is unable to be generated by other methods.

5.1. An overlap cluster

In order to test the effectiveness of the algorithm, both LDBSCAN and OPTICS are applied to a data set with 473 points as shown in Fig. 9. This set of data is also used in Fig. 7. The main result generated by LDBSCAN and OPTICS is the cluster formed by the pink points. Both LDBSCAN and OPTICS can generate the magenta cluster D, the cyan cluster E, and the green cluster F. But OPTICS can only generate the cluster G which contains all the magenta, cyan, green, and pink points. And it is more reasonable to generate a cluster which only contains the pink points because of their similarity in local-density.

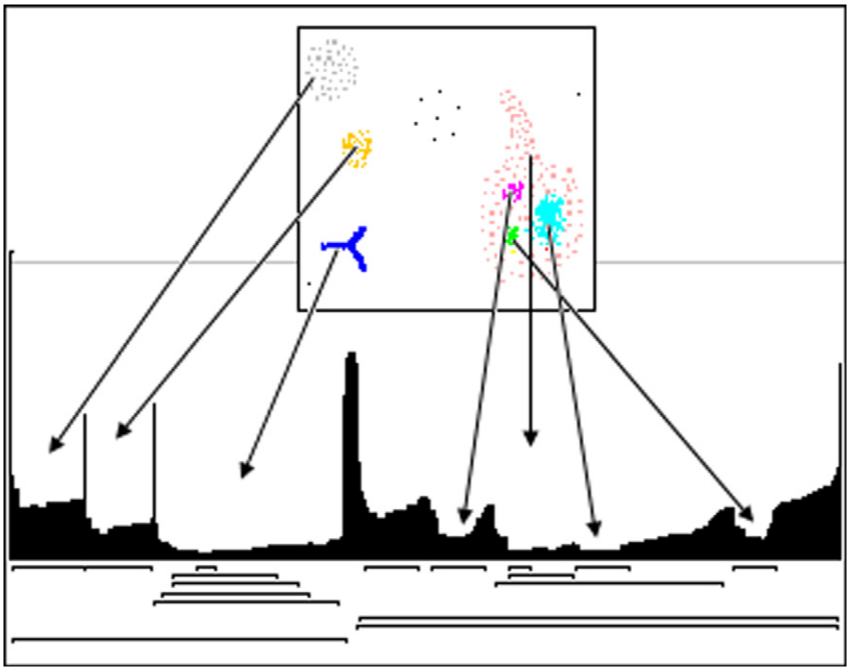

Fig. 9. An overlap cluster.

Therefore LDBSCAN produces the similar local-density clusters instead of the clusters produced by OPTICS with local-density exceeds certain thresholds. Of course, LDBSCAN can also generate the cluster G if we loosen the limit of pct as is shown in Fig. 7(b).

5.2. Performance

In this section, the performance of LDBSCAN is evaluated. The following experiments are conducted with a Pentium IV 2.4G computer with 512 MB main memory running Redhat Linux 9. All algorithms were implemented in Java by using Sun's JDK version 1.4.2. The data sets used were generated randomly, containing the statistical performance data of different core network nodes.

To cluster all the n objects in the database D, we implemented a three-step algorithm. In the first step, the MinPts-nearest neighborhoods are found, and in the second step all the LOFs and LRDs are computed, and then all the n objects are clustered in the third step.

The runtime complexity of the first and second steps has been carefully discussed in [1]. In the first step, the MinPts-nearest neighbors for every point p are materialized, together with their distances to p. The result of this step is a materialization database M of size $n * MinPts$ distances. Note that the size of this intermediate result is independent of the dimension of the original data. The runtime complexity of this step is $O(n*\text{time for a } k\text{-}nn \text{ query})$. For the k-nn queries, there is complexity of $O(n)$ for low-dimensional data, and complexity of $O(n \log n)$ for medium-dimensional data using X-tree [12] and complexity of $O(n^2)$ for high-dimensional data using VA-file [13]. In the second step all the LRD and LOF values are computed using the materialization database M and the time complexity of this step is $O(n)$.

In the third step, object p is assigned to a certain cluster or is marked as noise. The original database D is not needed for this step, as M contains sufficient MinPts-nearest neighbor information for LDBSCAN to generate clusters. Each point will be clustered only once, so the complexity of this step is $O(n)$. This is confirmed by the graph shown in Fig. 10.

So the run-time of the algorithm LDBSCAN is nearly the same as the run-time of LOF and the complexity of LDBSCAN is decided by the complexity of the first step k-nn query.

6. Conclusions

Density-based clustering algorithms are attractive for the task of class identification in spatial

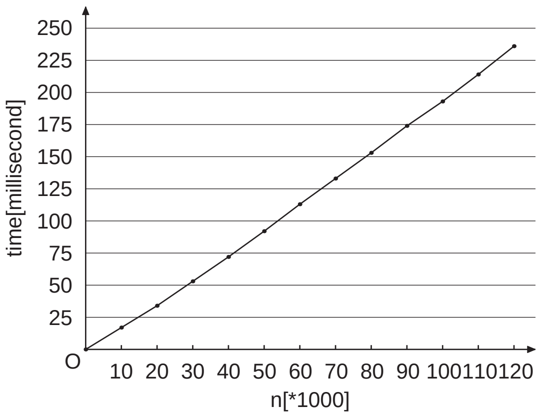

Fig. 10. Runtime for the clustering step.

database. However, they suffer from severe drawbacks as applied to cluster data with different local densities. In this paper, a clustering algorithm LDBSCAN which relies on a local-density-based notion of cluster is proposed. Experiments have been performed on synthetic data. The experiments show that the proposed algorithm provides better results than OPTICS and overcomes the shortcoming of DBSCAN. In addition it takes the advantage of the LOF to detect the noises compared with other density-based clustering algorithms.

There are several opportunities for future research. For example if LDBSCAN is started with different core points, there will be some border points that may be generated into different clusters. Now these border points are only assigned to the cluster discovered first. In fact these border points do not necessary belong to certain cluster and they can be assigned to several clusters simultaneously. In addition, the current acceptable local-density range is calculated according to the local-density of the current point. It might yield a more reasonable result if weights are assigned to the points as that assigned to the same cluster of the current point.

References

[1] M.M. Breunig, H.-P. Kriegel, R.T. Ng, J. Sander, LOF: identifying density-based local outliers, in: W. Chen, J.F. Naughton, P.A. Bernstein (Eds.), Proceedings of ACM SIGMOD International Conference on Management of Data, Dalles, TX, ACM, New York, 2000, pp. 93–104.

[2] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[3] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: E. Simoudis, J. Han, U.M. Fayyad (Eds.), Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Port Land, OR, AAAI, Menlo Park, CA, 1996, pp. 226–231.

[4] M. Ankerst, M.M. Breunig, H.-P. Kriegel, J. Sander, OPTICS: ordering points to identify the clustering structure, in: A. Delis, C. Faloutsos, S. Ghandeharizadeh (Eds.), Proceedings of ACM SIGMOD International Conference on Management of Data, Philadelphia, PA, ACM, New York, 1999, pp. 49–60.

[5] G. Sheikholeslami, S. Chatterjee, A. Zhang, WaveCluster: a multi-resolution clustering approach for very large spatial databases, in: A. Gupta, O. Shmueli, J. Widom (Eds.), Proceedings of 24th International Conference on Very Large Data Bases, New York, NY, Morgan Kaufmann, Los Altos, CA, 1988, pp. 428–439.

[6] H. Li, L. Xu, J. Wang, Z. Mo, Feature space theory in data mining: transformations between extensions and intensions in knowledge representation, Expert Syst. 20 (2) (2003) 60–71.

[7] H. Li, L. Xu, Feature space theory—a mathematical foundation for data mining, Knowledge-Based Syst. 14 (2001) 253–257.

[8] A. Hinneburg, D.A. Keim, An efficient approach to clustering in large multimedia databases with noise, in: R. Agrawal, P.E. Stolorz, G. Piatetsky-Shapiro (Eds.), Proceedings of Fourth International Conference on Knowledge Discovery and Data Mining, New York, NY, AAAI, Menlo Park, CA, 1998, pp. 58–65.

[9] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data for data mining applications, in: L.M. Haas, A. Tiwary (Eds.), Proceedings of ACM SIGMOD International Conference on Management of Data, Seattle, WA, ACM, New York, 1998, pp. 94–105.

[10] S. Ma, T.J. Wang, S.W. Tang, A New Fast Clustering Algorithm Based on Reference and Density, Lectures Notes in Computer Science, vol. 2762, Springer, Berlin, 2003, pp. 214–225.

[11] W.M. Ma, C. Eden, W.S. Tommy, A new shifting grid clusting algorithm, Pattern Recognition 37 (3) (2004) 503–514.

[12] S. Berchtold, D.A. Keim, H.-P. Kriegel, The X-tree: an index structure for high-dimensional data, in: T.M. Vijayaraman, A.P. Buchmann, C. Mohan, N.L. Sarda (Eds.), Proceedings of 22nd Conference on Very Large Data Bases, Bombay, India, Morgan Kaufmann, Los Altos, CA, 1996, pp. 28–39.

[13] R. Weber, H.-J. Schek, S. Blott, A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces, in: A. Gupta, O. Shmueli, J. Widom (Eds.), Proceedings of 24th International Conference on Very Large Data Bases, New York, NY, Morgan Kaufmann, Los Altos, CA, 1998, pp. 194–205.