The GridOPTICS clustering algorithm

Anikó Vágner, Faculty of Informatics, University of Debrecen,

26 Kassai Str, 4028 Debrecen, Hungary

email: vagner.aniko@inf.unideb.hu

Abstract

The OPTICS algorithm is a hierarchical density-based clustering method. It creates reachability plots to identify all clusters in the point set. Nevertheless, it has limitation, namely it is very slow for large data sets. We introduce the GridOPTICS algorithm, which builds a grid structure to reduce the number of data points, then it applies the OPTICS clustering algorithm on the grid structure. In order to get the clusters, the algorithm uses the reachability plots of the grid structure, then it determines to which cluster the original input points belong. The experimental results show that our new algorithm is faster than the OPTICS, the speed-up can be one or two orders of magnitude or more, which depends mainly on the τ parameter of the GridOPTICS algorithm. At the end of the article, we give some advice to which point set you can apply the GridOPTICS algorithm.

Keyword: clustering, large data set, OPTICS, grid

1. Introduction

Cluster analysis is one of the important research fields of data mining, which is applied on many other disciplines, such as pattern recognition, image processing, machine learning, bioinformatics, information retrieval, artificial intelligence, marketing, psychology, etc. Data clustering is a method of creating groups or clusters of objects in a way that objects in one cluster are very similar to each other and objects in different clusters are quite distinct. In data clustering, the classes is not predefined, clustering algorithms determine them. (Gan et al., 2007)

There are many effective clustering algorithms, such as grid-based, hierarchical, fuzzy, centre-based, search-based, graph-based, density-based, model-based, subspace clustering, etc. (Gan et al., 2007), (Han and Kamber, 2006). Considering the topic of our article, we introduce the grid-based and the density-based techniques. Both of them “are popular for mining clusters in a large multidimensional space wherein clusters are regarded as denser regions than their surroundings”. (Gan et al., 2007)

The grid-based clustering creates a grid structure in a way from the data points in the first step, in other words it partitions the data points into a finite number of cells and calculates the cell density for each cell. In the next step, the algorithm operates on the grid structure to identify the clusters (Gan et al., 2007). The great advantage of grid-based clustering is its significant reduction of the computational complexity, especially for clustering very large data sets, which means, its processing time is fast, because similar data points will belong to the same cell and will be regarded as a single point. “This makes the algorithms independent of the number of data points in the original data set.” (Gan et al., 2007). Well-known grid-based clustering techniques are the STING (Wang et al., 1997), the CLIQUE (Agrawal et al., 1998), the Wave-Cluster (Seikholeslami et al., 1998) and the OptiGrid (Hinneburg and Keim, 1999). Han and Kamber (2006) and Gan et al., (2007) gave a comprehensive summary of these techniques.

The density-based clustering approach is capable of finding arbitrarily shaped clusters. The clusters are dense regions, which are separated by sparse regions. These algorithms can handle noise very efficiently. “The number of clusters is not required as a parameter, since density-based clustering algorithms can automatically detect the clusters”, and in this way they determine the number of the clusters as well. There is a disadvantage of the most density-based techniques that it is hard to choose parameter values in order that the algorithm gives an appropriate result. (Gan et al., 2007) Han and Kamber (2006) reviewed the well-known density-based clustering algorithms, which are the DBSCAN (Ester et al., 1996), the DENCLUE (Hinneburg and Gabriel, 2007), and the OPTICS (Ankerst et al., 1999).

Clustering algorithms are sensitive to input parameters, in other words they have a significant influence on the results of clustering. It is not easy to find the parameters which ensure satisfying results. “The OPTICS algorithm 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.” (Ankerst et al., 1999)

On the other hand, the OPTICS clustering algorithm has also limitations, namely it has high complexity, which means that it is very slow for large data sets (Yue et al., 2007) (Schneider and Vlachos, 2013). In this paper, we introduce a new algorithm named GridOPTICS which uses a grid structure to reduce the number of data points and then applies the OPTICS algorithm on the grid structure to find the clusters. The new algorithm has the advantage that it has fast processing time, whereas it also keeps the advantages of the OPTICS.

The rest of the paper is organized as follows. Section 2 gives a short summary of new techniques of the grid-based and the density-based clustering and of their combinations. In Section 3, there is an overview of the main definitions of the OPTICS algorithm. In Section 4, each step of our GridOPTICS algorithm is described, whereas Section 5 introduces how we implemented the algorithms. Section 6 gives the experimental results, namely we compare the execution time, the reachability plots and the clustered points of the OPTICS and the GridOPTICS for more point sets. Before the conclusion, we give advice when and how to apply our application. At the end, we summarize our results.

2. Related works

Combination of the grid-based and the density-based technique is common. Parikh and Varma (2014) gave a short survey of this topic. They presented short descriptions of some grid-based algorithm namely the new shifting clustering algorithm, the grid-based DBSCAN algorithm, the GDILC algorithm, the general grid-clustering approach, and the OPT-GRID(S).

Similarly, Mann and Kaur (2013) collected some DBSCAN variant algorithms, namely GMDBSCAN and GDCLU combined the two techniques. The grid-based DBSCAN algorithm (Darong and Peng, 2012) is similar to our algorithm; however, they improved the DBSCAN algorithm.

G-DBSCAN (Ma et al., 2014) uses a grid method for the first time, and removes noise in order to reduce the points to be processed. Its goal was to reduce memory usage and improve efficiency of the algorithm. They did not give exact information how they assign input points to the grid structure, moreover, they analysed the efficiency of their algorithm on data sets which have only about a few hundred points.

Zhao et al. (2011) proposed an enhanced grid-density based approach for clustering high dimensional data, which was accurate and fast, which they showed in the experimental evaluation, where they executed their AGRID+ algorithm for more synthetic data sets. Ma et al. (2003) presented the CURD algorithm, which uses references and density, and which has nearly linear time complexity. Achtert et al. (2006) introduced the DeLiClu algorithm, which avoids the non-intuitive ε parameter of the OPTICS and the density estimator of single-link algorithm.

Some researchers changed the OPTICS algorithm in order to improve it in a way. Schneider and Vlachos (2013) introduced the fast density-based clustering technique based on random projections (FOPTICS), whose goal was to speed up the computation of the OPTICS algorithm. Alzaalan et al. (2012) enhanced the concept of core-distance of the OPTICS in order to make the algorithm less sensitive to data with variant density but they did not improve the performance. Patwary et al. (2013) introduced the scalable parallel OPTICS algorithm (POPTICS), which they tried out on a 40-core shared-memory machine. Brecheisen et al. (2006) confined the OPTICS algorithm to ε-range queries on simple distance functions and carried out complex distance computations only at a stage of the algorithm where they were compulsory to compute the correct clustering result. Breunig et al. (2000) combined compression, namely the BIRCH algorithm, with the OPTICS algorithm to yield performance speed-up-factors. Brecheisen et al. (2006) combined the multi-step query processing with density-based clustering algorithms, namely with the DBSCAN and the OPTICS in order to accelerate them by more than one order of magnitude.

Yue et al. (2007) presented a new algorithm named OGTICS, which modifies and improves the OPTICS with grid technology and has linear complexity, thus it is much faster than the OPTICS. The algorithm divides the data set into number of grids and assigns all data into these grids, then it partitions all grids to a few groups. In the next steps, it computes the centre of each grid and orders all grids to a queue in x-axis. In the last two steps, it generates a statistical histogram with the number of data across all grids and determines the optimal number of clusters and partitions.

Our algorithm differs from this algorithm in creating of the grid structure and in processing of the grid structure. Namely, our algorithm builds a simpler grid structure, and we use the OPTICS algorithm with only a few changes in the processing, whereas they used ordering the grids by x-axis.

3. The OPTICS algorithm

The OPTICS (Ankerst et al., 1999) has two input parameters (ε and MinPts) which are used to find neighbours, core objects, core-distances, and reachability-distances. They introduced some other definition but only the previous definitions are important to describe our algorithm.

The *neighbours* of the C point are the points which are in the ε-neighbourhood of the C. C point is a *core-object* if the cardinality of the ε-neighbourhood of the C point is equal or greater than MinPts. The *core-distance* of the C point is the smallest ε' (ε'<= ε) of which it is true that the cardinality of the ε'-neighbourhood of the C point is equal or greater than MinPts; if this ε' does not exists, it is undefined. The *reachability-distance* of P point with regard to C point is undefined if the C is not core-object, otherwise the greater value from the core-distance of C point and the distance of the P and the C points.

The OPTICS algorithm generates a structure in which the sequence of the input points is important, and it assigns a corresponding reachability-distance for each point. This structure can be displayed by 2-D plots, the name of which is reachability plots. Valleys in reachability plots indicate

clusters: points having a small reachability value are closer and thus more similar to their predecessor points than points having high reachability value. (Brecheisen et al., 2006)
If you want to determine the clusters of this structure, you can use the algorithm of Ankerst et al. (1999). However, Patwary et al. (2013) provided a simpler algorithm. Both of them need a new φ parameter ($0<=\varphi<=\varepsilon$), and they consider a cluster as an interval, where is it true that reachability distance of every point of the interval is not greater than φ. The cardinality of a cluster should be at least MinPts.

4. The GridOPTICS algorithm

The main idea of the GridOPTICS algorithm is to reduce the number of input points with a grid technique and then to execute the OPTICS algorithm on the grid structure. Based on the reachability plots, the clusters of the grid structure can be determined. In the end, the input points can be assigned to the clusters. It is supposed that the points are in the Euclidean space, so the Euclidean distance is used, however other distances can also be used. The GridOPTICS algorithm has 3 parameters, namely ε, MinPts, which play similar role as in the OPTICS, and τ, which defines the distance in the grid structure.
The algorithm has 4 main steps, they are the following:

1. Step: Constructing the grid structure

The grid structure is very simple, that is there are grid lines which are parallel and their distance is τ in a dimension, moreover they are orthogonal to each other if they are not in the same dimension. In this way, they cross each other in grid points. The distance of two neighbour grid points is τ.
In an n-dimensional space, an input point (p_1, p_2, … p_n) is assigned to a grid point (g_1, g_2, … g_n) (g_i=k, k is element of integers, i=1, …, n) in the following way: $g_i-\tau/2 <=p_i <g_i+ \tau/2$, (i=1, …, n). The algorithm counts how many input points belong to each grid point. It only stores the grid points to which at least one input point has been assigned.
Figure 1 shows a simple example how the algorithm assigns the input points to the grid in two dimensions. On the left side, there are the input points and the grid lines, on the right side, there are the grid points and the number which shows how many input points belong to each grid point.
In this way, each input point is transformed into a grid point, which can hurt accuracy in a minimal way. If the grid was moved with τ’ ($-\tau <\tau'< \tau$) in a dimension, an input point would be likely to be transformed into another

grid point but the inaccuracy problem would be the same. You will see that it could influence the results only to the slightest degree.

2. Step: Applying the OPTICS algorithm to the grid structure

The OPTICS searches the points in the ε-neighbourhood of a point more times. To perform this task it should examine all input points. Because of the grid, this task is simpler than in the OPTICS, since the neighbours of a grid point are also in the grid structure. We know that the distance of two neighbour grid points is τ, and we want to find points in ε-neighbourhood of a point. Figure 2 shows the neighbour grid points of the C. The serial numbers of the grid points show the order in which the algorithm should process them when it calculates the core-distance of the C point.

Zhao et al. (2011) gave a comprehensive discussion about the neighbourhood on a high-dimensional grid structure.

In the second step, the algorithm calculates the core-distance of each stored grid point (C point) firstly. The OPTICS defines the core-distance of C point as the smallest ε'<= ε of which it is true that the cardinality of the ε'-neighbourhood of the C point is equal or greater than MinPts; if this ε' does not exists, it is undefined. The GridOPTICS algorithm calculates it in the next way:

1. if the number of the points assigned to the C is more than MinPts, the core-distance will be 0;
2. if the distance of the C and the points marked by 1 on Figure 2 (which is τ) is not more than ε, and the number of input points assigned to the C and the grid points marked by 1 is not less than MinPts, the core-distance will be the distance of the C and the points marked by 1 (in this case this is τ);
3. if the distance of C and the points marked by I (I = 2, 3, …, ε/τ) on Figure 2 is not more than ε, and the number of input points assigned to the C and the grid points marked by 1, 2, … I is not less than MinPts, the core-distance will be the distance of the C and points marked by I;
4. otherwise the core-distance is undefined.

The other part of the second step is almost the same as the steps of the OPTICS. The algorithm chooses a grid point from the unprocessed grid points, and accounts it processed, then if it is a core-object, the algorithm continues the processing with the neighbour grid points, otherwise the algorithm repeats this step until there are unprocessed grid points.

In the processing of the neighbour grid points, the algorithm searches the neighbour grid points and puts them into a neighbour collection. Then, it chooses the point of the collection which has the smallest reachability-distance, accounts it processed, takes out from the neighbour collection, and if the point is core-object, the algorithm adds its neighbour grid points to the neighbour collection. Until the neighbour collection is not empty, the algorithm continues the processing of the next element of the neighbour collection.

Figure 3 shows the pseudo-code of the second main step.

As a result, there is a structure in which there is a given sequence of grid points with their corresponding reachability-distances.

3. Step: Determining clusters of the grid points

In this step, the algorithm assigns a cluster number to each cluster. We do not find automatically all clusters as Ankerst et al. (1999), instead we follow the method of Brecheisen et al. (2006), moreover our algorithm is similar to the algorithm of Patwary et al. (2013) but it is not the same. We also used the results of Sander et al. (2003), who automatically determined the significant clusters in reachability plots with the help of dendograms.

Our goal is to find the clusters in which the reachability distance is less than φ (0<=φ<=maximum of the reachability distances, or φ is undefined). We could also find all clusters if we applied all φ values but we will show results only for a few φ values. However, if you need all clusters, you can apply the algorithm of Ankerst et al. (1999) on the processed grid points of the GridOPTICS.

Our algorithm processes the sequence of the grid points, and it says that if the reachability distance of a point is bigger than φ, there is a new cluster. However, if a cluster has fewer grid points than MinPts, it is noise, so it should examine how many points the cluster under processing has.

Figure 4 shows the pseudo-code of the third step.

4. Step: Assigning the input points to the clusters

In the last step, the GridOPTICS algorithm determines to which cluster each input point belongs. It looks through the input points and searches the grid point which was assigned to it, and reads its cluster number.

If you want to find all clusters in the reachability plots, you should execute the last two steps for all φ values.

5. Implementation

We realized the algorithm in the C# language in the Visual Studio 2010 Express Edition. We executed our program on 2-dimensional input points.

We created the grid structure in a way that we shifted the input points in order that the minimum coordinates could be 0, and we divided the coordinates by τ. Consequently, the indexes of the grid structure started with 0 and its coordinates are small non-negative integer values, which makes the calculation faster.

In the core-distance calculation, we could use that the neighbour grid points of C (cx,cy) marked by 1 on Figure 2 are (cx-1,cy), (cx+1,cy), (cx,cy-1), (cx, cy+1). Similarly, it is very easy to find the coordinates of the other neighbour grid points. To make the search of the neighbours easy, we used a List collection to store the relative coordinates of the possible ε-neighbours in the appropriate order shown in Figure 2, and we stored the grid structure in a Dictionary collection, which supported the search by coordinates.

The neighbour and the processed grid points were stored in List collections because the algorithm did not need to search the grid points by coordinates any more, but it needed the order of the processed grid points to produce the reachability plots.

6. A basic example

We examine the steps of the GridOPTICS algorithm with a synthetic point set which has 20 points. Table 1 shows the input points named PointSetFirstTry20. The parameters of the algorithms are ε = 50, MinPts=3, τ=4.

In the first step, the GridOPTICS algorithm creates the grid structure, which is shown in Table 2. There are the two coordinates of the grid structure in the first two columns, whereas the third column shows how many input points belong to the grid point.

In the second step, firstly the algorithm calculates the core distances, which is shown in the fourth column. The core distances are calculated for the input points, namely the distances of the grid points are multiplied with the value of τ. Then, the algorithm orders the points with the help of their reachability distances. The fifth column of Table 2 shows the reachability distances; moreover, the sequence of the grid points in Table 2 corresponds to the result of the algorithm. This means that the sequence of the grid points and their reachability distances constitute the reachability plots.

In the third step the algorithm assigns a cluster number to each clusters of grid points. The sixth and seventh columns of Table 2 show the cluster numbers when φ=8 and φ=22.

In the last step, the algorithm determines to which cluster each input point belongs. The third and fourth columns of Table 1 show the cluster numbers of input points. The order of the input points is the same as the original order, the reachability plots cannot be read from there.

7. Experimental Results

Firstly, we used some home-generated synthetic point sets as input point sets of our GridOPTICS algorithm. The coordinates of the input points are integer values. The synthetic point sets have noisy points in order to show how the GridOPTICS finds them. Moreover, they have more clusters and these clusters have various densities in order that there are more valleys of the reachability plots. The goal of the valleys is that we could easier make comparison between the reachability plots resulted by OPTICS and GRIDOPTICS. The program is executed on a PC which had 2GB RAM, and 2.01 GHz AMD Athlon CPU.

We make comparison of execution time and results of the OPTICS and the GridOPTICS algorithm. Brecheisen et al. (2006) state that "ε has to be very high in order to create reachability plots without loss of information and MinPts is typically only a small fraction of cardinality of input data, e.g., MinPts=5 is a suitable value even for large databases". In the experience, we used very large ε values in order to easily compare the reachability plots; moreover, we also used higher MinPts values.

The following abbreviations are used in the tables: OT is the execution time of the OPTICS, GOT is the execution time of the GridOPTICS, NGP is the number of the grid points, and MP is MinPts. The measure of the execution time is millisecond.

First, we executed both algorithms on a synthetic point set with 407 input points called PointSet500. The x coordinates range between 84 and 573, whereas y coordinates are between 410 and 815. Table 3 shows the execution time of the OPTICS and the GridOPTICS and the number of grid points of the GridOPTICS.

If τ=1, the grid structure is almost the same as the input points, so the GridOPTICS will have almost the same result as the OPTICS but the execution will be longer because the GridOPTICS builds the grid structure first. If τ is bigger, the grid structure will have fewer grid points, so it will be faster. You can read from Table 3, if τ >=10, the execution time will be less with one or two orders of magnitude or more.

Figure 5 shows the results executed on the PointSet500 with ε=500, MinPts=5. Subfigure A shows the reachability plots produced by the OPTICS, Subfigure B is the clustered points resulted the OPTICS where φ=44. The C, D, E, F, G and H parts of the figure show the reachability plots produced by the GridOPTICS, where τ=1, 5, 10, 20, 30, 40. The red line in reachability plots of each subfigures shows the value of the φ, which is 44 in these cases.

You can see little green points on the reachability plots of the GridOPTICS in Figure 5, which show how many input points belong to a grid point. The reachability plots of the GridOPTICS executed with τ=1 parameter are similar to that of the OPTICS, the algorithm can produce almost the same results as the OPTICS. The other reachability plots show us that if the τ is higher, the plot is plainer, there are not so many cuts in the valleys, moreover, more input points belong to a grid point, consequently the reachability distances in the valleys are growing with the τ.

In case of this point set the τ=40 is very high because it means that there are 10-12 gridlines in each dimension. This can cause inaccuracy, namely Figure 6 shows its result, where you see two points in two red circles which should be noise instead of clustered points.

The GridOPTICS with the other τ values created the same clusters as the OPTICS in the above described cases.

Figure 7 and 8 show reachability plots of PointSet500.

The GridOPTICS can cause inaccuracy because the original input points are substituted with grid points. The higher the τ is, the more inaccurate the result is. This inaccuracy can cause that a few input points are clustered but they should be noise, or the GridOPTICS consider some input points to be noise but they should be clustered. In the most cases, these input points are in the border of a cluster. Another inaccuracy can happen when two or more clusters are contracted.

The grid structure may be moved with τ‘ (-τ< τ’ < τ) in a dimension but it will cause the same inaccuracy on the other side of the clusters.

With higher MinPts value, the reachability plots of the OPTICS are also plainer, there are not so many cuts in the valleys. In this way, the reachability plots of the GridOPTICS can be more similar to it. We can state if the MinPts is higher, the GridOPTICS will be less or not inaccurate.

Let us see other input points. PointSet1000 has 919 synthetic points, the x coordinates range between 234 and 572, whereas y coordinates are between 390 and 783. Table 4 shows the execution time of the GridOPTICS and the OPTICS on the PointSet1000.

In case of this point set, there will be less than 10 gridline in each dimension if you choose τ >=30. In this way, there will not be enough grid points in order that the GridOPTICS can give an accurate result.

Part A of Figure 9 shows clustered points and the C part shows the reachability plots resulted by the OPTICS with ε =400, MinPts=5 and φ =20. Subfigure B demonstrates clustered points, whereas Subfigure D displays the reachability plots of the GridOPTICS with ε =400, MinPts=5, τ=10, and φ =20. In this case, the clusters are similar to each other, but on Subfigure B you can find inaccuracy, namely you can find input points which are noise instead of being clustered. These are caused by the transformation into the grid structure. At the same time, the GridOPTICS is thirty-four times faster than the OPTICS in this case.

PointSet4000 has 4028 synthetic points; the x coordinates range between 37 and 986, whereas y coordinates are between 20 and 933. Table 5 shows the execution time of the algorithms on this point set.

The OPTICS takes about 1,5 hour to give results for a set which has about 4000 points, whereas the GridOPTICS takes about 24 minute if τ=5, it takes 2 minutes if τ=10, and it takes less than a minute if τ>=20.

Part A of Figure 10 shows reachability plots, Subfigure C and E display the clustered points of the OPTICS with ε =1000, MinPts=20, φ =25 and φ =45, in case of Subfigure C φ =25, whereas on Subfigure E φ =45. Subfigure B shows reachability plots, whereas D (φ =25) and F (φ =45) parts show the clustered points of the GridOPTICS with ε =1000, MinPts=20, τ=20, φ =25 and φ =45.

In this case, the GridOPTICS is faster about 400 times as the OPTICS but the GridOPTICS loses information, namely if φ=25, the GridOPTICS finds a lot of noise (noise points are black on the figures), where the OPTICS finds clusters, furthermore the GridOPTICS contracts clusters together. However, the GridOPTICS finds the main clusters similarly to the OPTICS. If φ =45, the results of the two algorithms are fast the same.

PointSet5000 has 5045 synthetic points, the x coordinates range between 22 and 978, whereas y coordinates are between 16 and 934. Table 6 shows the execution time of the algorithms on this point set.

Considering the execution time it took the OPTICS algorithm about three hours to give results, whereas it took the GridOPTICS about 7 minutes when τ =10.

Figure 11 shows the results executed on the PointSet5000 with ε=1000 and MinPts=5. Subfigure A shows the reachability plots resulted by the OPTICS and Subfigure B shows the reachability plots resulted by the GridOPTICS with τ =10. φ=12 and φ=32 which values are represented by the red lines on these two subfigures. Subfigure C and E show the clustered points resulted by the OPTICS, in case of the C, φ=12, whereas in case of the E, φ=32. Similarly, Subfigures D and F show the clustered points resulted by the GridOPTICS, in case of the D, φ=12, whereas in case of the F, φ=32.

You can see a similar inaccuracy on Figure 11 as Figure 10, namely the GridOPTICS finds a lot of noise, where the OPTICS finds clusters, and the

GridOPTICS contracts clusters together. At the same time, the execution time of the GridOPTICS is 23 times faster than the OPTICS in this case.

PointSet3000 has 2976 synthetic points; the x coordinates range between 204 and 877, whereas y coordinates are between 60 and 876. Table 7 shows the execution time of the algorithms on this point set.

Figure 12 shows the results executed on PointSet3000 with ε=800, MinPts=5. Subfigure A shows the reachability plots resulted by the OPTICS, whereas B, C and D parts show the reachability plots resulted by the GridOPTICS with τ =5, 10 and 20. φ=6 and φ=21 are chosen which are represented by the red lines on each subfigure. Subfigure A, B, C, and D are cut and enlarged in order that you can see the important parts of the reachability plots. The E and G parts of Figure 13 show the clustered points resulted by the OPTICS in case of the E φ=6, whereas in case of the G φ=21. Similarly, F and H parts of the figure show the clustered points resulted by the GridOPTICS τ =5, in case of the F φ=6, whereas in case of the H φ=21. Subfigure I and J show the clustered points resulted by the GridOPTICS, in case of I, τ =10 and φ=21, whereas in case of J τ =20 and φ=21. In case of the GridOPTICS with τ =10 and 20 and φ=6, all input points are noises.

You can see on Figure 12 and Figure 13 that it is not worth choosing higher value for the τ as 10 in this case, because the clusters having small granularity are lost. Therefore, we can suggest using lower τ value, namely τ =5. The cardinality of the point set is slight, consequently the execution time is not so long.

We also executed both algorithms on some clustering data sets from the http://cs.joensuu.fi/sipu/datasets/ website. Figure 14 shows the results executed on the Aggregation data set (Gionis, 2007), where Table 8 shows the execution time of the algorithms. The Aggregation data set has 788 points, the x coordinates range between 335 and 3655, whereas y coordinates are between 195 and 2915.

Figure 15 shows the results executed on the Dim2 data set, where Table 9 shows the execution time of the algorithms. The Dim2 data set has 1351 points, the x coordinates range between 0 and 978207, whereas y coordinates are between 0 and 1000000.

Figure 16 shows the results executed on the A1 data set (Kärkkäinen and Fränti, 2006), where Table 10 shows the execution time of the algorithms. The A1 data set has 3000 points, the x coordinates range between 0 and 65535, whereas y coordinates are between 32064 and 64978.

Figure 17 shows the results executed on the S3 data set (Fränti and Virmajoki, 2006), where Table 11 shows the execution time of the algorithms. The S3 data set has 5000 points, the x coordinates range between 32710 and 942327, whereas y coordinates are between 70003 and 947322.

Figure 18 shows the results executed on the Unbalance data set, where Table 12 shows the execution time of the algorithms. The Unbalance data set has 6500 points, the x coordinates range between 139779 and 575805, whereas y coordinates are between 271530 and 440940.

Figure 19 shows the results executed on the t4.8k data set (Karypis et al., 1999), where Table 13 shows the execution time of the algorithms. The t4.8k data set has 8000 points, the x coordinates range between 14642 and 634957, whereas y coordinates are between 21381 and 320874.

Figure 20, 21 and 22 show examples for the results of the GridOPTICS executed on data sets with 100000 and 50000 data points.

8. Application of the GridOPTICS

We suggest using the GridOPTICS if you have more than 500-600 input points, because if you have only less than 500 input points, the OPTICS works fast, it takes only 1 second to give results. But, you can read from Table 4 (PointSet1000) that if you have about 1000 points, it takes the OPTICS about 1,5 hour to give results on a simple PC, whereas it takes the GridOPTICS 13 second if τ=5.

Moreover, the GridOPTICS is very useful, if you have a data set where a lot of input points have the same coordinates with each other, which means that the number of the grid points may be a fractional part of the number of input points even if τ=1. Figure 23 shows an example for such a data set, where τ=1 and the cardinality of the grid points is fewer with one order of magnitude as the cardinality of input points.

In most examples, we used large ε values in order to show entire reachability plots. Of course, you can execute the GridOPTICS with smaller ε, which will results that it may not find the rough-grained clusters. The OPTICS works similarly. In Figure 24 you can find an example for smaller ε, namely they are the reachability plots and the clustered points of the GridOPTICS on PointSet5000 with ε=40, MinPts=5, τ =10, and φ=32.

The τ should be less or equal with ε, otherwise a cluster will consist of the input points which belong to a grid point. Of course, it can also be considered a clustering algorithm.

We would estimate the value of τ based on the cardinality of the point set and the range of the coordinates. We would choose the τ in order to be at least 500-1000 grid points. If there are less than 1000 grid points, the GridOPTICS gives results in a few minute. However, 500-1000 grid points are not enough for large data sets. Hence, we could consider the relationship between the execution time and the cardinality of the grid points, because the execution time substantially depends on the cardinality of the grid points. Finally, who will apply this algorithm, should decide what they need. On the one hand, you can set a higher value to τ, consequently the algorithm will be faster but less accurate, however in case of large data sets the high τ values may scarcely cause accuracy problems. On the other hand, you can choose lower value for τ, which will result in longer execution time, but more accurate results.

9. Future Work

We plan to try out the algorithm for more dimensions and with more types of distances. One type of real world examples can be that the input data are GPS coordinates, which need a special distance.

We also plan to find an appropriate method to measure the quality of the GridOPTICS algorithm considering the OPTICS. Both algorithms generate reachability plots and it would be better if we could compare the reachability plots instead of the resulted clusters. But, we cannot find any quality measurement technique for the reachability plots. We plan to give one in the future.

10. Conclusion

In this paper, we introduced the GridOPTICS clustering algorithm. It builds a grid structure from the input points in the first step, then it executes the OPTICS algorithm on the grid structure in the second step, afterwards it determines the clusters of the grid points in the third step, and finally, it assigns the input points to the clusters. The algorithm has the same advantage as the OPTICS, namely it builds reachability plots to find more than one clustering structure. Moreover, its execution time can be faster with more orders of magnitude than the OPTICS, which is very useful for large data sets which have more thousand points. However, the GridOPTICS can be less accurate than the OPTICS. The user of our algorithm has to decide whether the accuracy or the speed of their application is important. In the experimental results, we made comparison between the execution times and results of the GridOPTICS and the OPTICS. At the end of the article, we gave suggestions when to use the GridOPTICS algorithm and how its parameter can be chosen, and finally, we gave more examples for data sets clustered by the GridOPTICS.

Acknowledgements

The publication was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.

The author thanks her PhD supervisor, Márton Ispány for his valuable contribution.

References

Achtert, Elke, Christian Böhm, and Peer Kröger. "DeLi-Clu: Boosting Robustness, Completeness, Usability, and Efficiency of Hierarchical Clustering by a Closest Pair Ranking." In *Advances in Knowledge Discovery and Data Mining*, edited by Wee-Keong Ng, Masaru Kitsuregawa, Jianzhong Li, and Kuiyu Chang, 3918:119–28. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006. http://dx.doi.org/10.1007/11731139_16.

Agrawal, Rakesh, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications." *SIGMOD Rec.* 27, no. 2 (June 1998): 94–105. doi:10.1145/276305.276314.

Alzaalan, Mahmoud E., Raed T. Aldahdooh, Wesam Ashour. "EOPTICS Enhancement Ordering Points to Identify the Clustering Structure". In *International Journal of Computer Applications (0975 – 8887)* Vol. 40 No.17, 2012

Ankerst, Mihael, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. "OPTICS: Ordering Points to Identify the Clustering Structure." In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, 49–60. SIGMOD '99. New York, NY, USA: ACM, 1999. doi:10.1145/304182.304187.

Brecheisen, Stefan, Hans-Peter Kriegel, and Martin Pfeifle. "Multi-Step Density-Based Clustering." *Knowl. Inf. Syst.* 9, no. 3 (March 2006): 284–308. doi:10.1007/s10115-005-0217-6.

Breunig, Markus M., Hans-Peter Kriegel, and Jörg Sander. "Fast Hierarchical Clustering Based on Compressed Data and OPTICS." In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, 232–42. PKDD '00. London, UK, UK: Springer-Verlag, 2000. http://dl.acm.org/citation.cfm?id=645804.669672.

Darong, Huang, and Wang Peng. "Grid-Based DBSCAN Algorithm with Referential Parameters." *Physics Procedia* 24, Part B, no. 0 (2012): 1166–70. doi:http://dx.doi.org/10.1016/j.phpro.2012.02.174.

Ester, Martin, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.” *KDD* 96, no. 34 (1996): 226–31.

Fränti, P. and Virmajoki, O., "Iterative shrinking method for clustering problems", Pattern Recognition, 39 (5), 761-765, May 2006.

Gan, Guojun, Chaoqun Ma, and Jianhong Wu. *Data Clustering: Theory, Algorithms, and Applications*. Vol. 20. SIAM, 2007.

Gionis, A., H. Mannila, and P. Tsaparas, Clustering aggregation. ACM Transactions on Knowledge Discovery from Data (TKDD), 2007. 1(1): p. 1-30.

Han, Jianwei, Micheline Kamber. Data Mining. Concepts and Techniques. *Elsevier*, 2006.

Hinneburg, Alexander, and Daniel A. Keim. “Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering.” In *Proceedings of the 25th International Conference on Very Large Data Bases*, 506–17. VLDB ’99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999. http://dl.acm.org/citation.cfm?id=645925.671505.

Hinneburg, Alexander, and Hans-Henning Gabriel. “DENCLUE 2.0: Fast Clustering Based on Kernel Density Estimation.” In *Advances in Intelligent Data Analysis VII*, edited by Michael R. Berthold, John Shawe-Taylor, and Nada Lavrač, 4723:70–80. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007. http://dx.doi.org/10.1007/978-3-540-74825-0_7.

Kärkkäinen, Ismo and Fränti, Pasi. "Dynamic local search algorithm for the clustering problem", Research Report A-2002-6

Karypis, G., Han, E.H., Kumar, V.. CHAMELEON: A hierarchical 765 clustering algorithm using dynamic modeling, IEEE Trans. on Computers, 32 (8), 68-75, 1999.

Ma, Li, Lei Gu, Sou yi Qiao, Jin Wang. ”G-DBSCAN: An Improved DBSCAN Clustering Method Based On Grid.” In *Advanced Science and Technology Letters* Vol.74 (ASEA), 2014: 23-28 http://dx.doi.org/10.14257/astl.2014.74.05

Ma, Shuai, TengJiao Wang, ShiWei Tang, DongQing Yang, and Jun Gao. “A New Fast Clustering Algorithm Based on Reference and Density.” In *Advances in Web-Age Information Management*, edited by Guozhu Dong, Changjie Tang, and Wei Wang, 2762:214–25. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003. http://dx.doi.org/10.1007/978-3-540-45160-0_21.

Mann, Amandeep Kaur, Navneet Kaur. “Grid Density Based Clustering Algorithm.” In *International Journal of Advanced Research in Computer Engineering & Technology* (IJARCET) Vol. 2, Issue 6, 2013: 2143-2147.

Parikh, Monali, Tanvi Varma. “Survey on Different Grid Based Clustering Algorithm.” In *International Journal of Advance Reseach in Computer Science and Management Studies.* Vol. 2, Issue 2, 2014. www.ijarcsms.com

Patwary, Mostofa Ali, Diana Palsetia, Ankit Agrawal, Wei-keng Liao, Fredrik Manne, and Alok Choudhary. “Scalable Parallel OPTICS Data Clustering Using Graph Algorithmic Techniques.” In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 49:1–49:12. SC ’13. New York, NY, USA: ACM, 2013. doi:10.1145/2503210.2503255.

Sander, Jörg, Xuejie Qin, Zhiyong Lu, Nan Niu, and Alex Kovarsky. “Automatic Extraction of Clusters from Hierarchical Clustering Representations.” In *Proceedings of the 7th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, 75–87. PAKDD ’03. Berlin, Heidelberg: Springer-Verlag, 2003. http://dl.acm.org/citation.cfm?id=1760894.1760906.

Schneider, Johannes, and Michail Vlachos. “Fast Parameterless Density-Based Clustering via Random Projections.” In *Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management*, 861–66. CIKM ’13. New York, NY, USA: ACM, 2013. doi:10.1145/2505515.2505590.

Sheikholeslami, Gholamhosein, Surojit Chatterjee, and Aidong Zhang. “WaveCluster: A Wavelet-Based Clustering Approach for Spatial Data in Very Large Databases.” *The VLDB Journal* 8, no. 3–4 (February 2000): 289–304. doi:10.1007/s007780050009.

Wang, Wei, Jiong Yang, and Richard R. Muntz. “STING: A Statistical Information Grid Approach to Spatial Data Mining.” In *Proceedings of the 23rd International Conference on Very Large Data Bases*, 186–95. VLDB ’97. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. http://dl.acm.org/citation.cfm?id=645923.758369.

Yue, Shihong, Miaomiao Wei, Yi Li, and Xiuxiu Wang. “Ordering Grids to Identify the Clustering Structure.” In *Advances in Neural Networks – ISNN 2007*, edited by Derong Liu, Shumin Fei, Zengguang Hou, Huaguang Zhang, and Changyin Sun, 4492:612–19. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007. http://dx.doi.org/10.1007/978-3-540-72393-6_73.

Zhang, Tian, Raghu Ramakrishnan, and Miron Livny. “BIRCH: A New Data Clustering Algorithm and Its Applications.” *Data Mining and Knowledge Discovery* 1, no. 2 (1997): 141–82. doi:10.1023/A:1009783824328.

Zhao, Yanchang, Jie Cao, Chengqi Zhang, and Shichao Zhang. “Enhancing Grid-Density Based Clustering for High Dimensional Data.” Journal of Systems and Software 84, no. 9 (2011): 1524–39.

Tables

Original x coordinate	Original y coordinate	Cluster Number if φ=8	Cluster Number if φ=22
54	7	1	1
2	5	noise	noise
30	3	noise	1
52	3	1	1
52	5	1	1
51	5	1	1
28	19	2	1
27	18	2	1
27	19	2	1
27	18	2	1
23	16	2	1
92	33	3	2
90	34	3	2
96	35	3	2
97	34	3	2
91	36	3	2
96	33	3	2
96	31	3	2
92	34	3	2
95	35	3	2

Table 1 – The FirstTry20 point set

X	Y	Card. of input points	Core Distance	Reach. Distance	Cluster Number if φ=8	Cluster Number if φ=22
13	1	1	5,656854	3,402823E+38	1	1
12	0	3	0	5,656854	1	1
7	0	1	16,49242	20	noise	1
5	3	1	5,656854	14,4222	2	1

6	4	4	0	5,656854	2	1
0	0	1	28	28	noise	noise
22	8	4	0	45,60702	3	2
23	8	1	4	4	3	2
24	8	3	0	4	3	2
24	7	1	4	4	3	2

Table 2 – The Grid points with the cardinality of the input points, the reachability distances, the core distances and the cluster numbers generated from the FirstTry20 point set (ε=50, MinPts=3, τ =4)

ε	MP	OT	τ	1	5	10	20	30	40
			NGP	391	225	100	55	39	18
500	5	6270	GOT	9392	2122	166	84	34	33
500	10	5583	GOT	9598	1512	211	76	11	6
500	20	6690	GOT	9598	1367	202	57	9	6

Table 3 – The execution time of the algorithms on the PointSet500

ε	MP	OT	τ	1	5	10	20	30
			NGP	863	484	241	118	76
400	5	64330	GOT	78660	13969	1849	266	86
500	10	69599	GOT	82527	14987	1648	290	64
400	20	61515	GOT	87628	16361	1759	245	97

Table 4 - The execution time of the algorithms on the PointSet1000.

ε	MP	OT	τ	5	10	20	30
			NGP	2365	1120	486	309
1000	5	5474880	GOT	1518406	164218	14750	3334
1000	10	5699831	GOT	1679268	175580	14774	3998
1000	20	5429290	GOT	1483909	159955	13513	3542

Table 5 - The execution time of the algorithms on the PointSet4000.

ε	MP	OT	τ	5	10	20	30
			NGP	3282	1589	646	408
1000	5	10853077	GOT	4388269	466100	32677	8193

Table 6 - The execution time of the algorithms on the PointSet5000.

ε	MP	OT	τ	5	10	20
			NGP	1666	953	434
800	5	2154836	GOT	550441	96625	10150

Table 7 - The execution time of the algorithms on the PointSet3000.

ε	MP	OT	τ	110
			NGP	399
3000	5	12696	GOT	2237

Table 8 - The execution time of the algorithms on the Aggregation.

ε	MP	OT	τ	10000
			NGP	145
1000000	5	70651	GOT	212

Table 9 - The execution time of the algorithms on the Dim2.

ε	MP	OT	τ	500
			NGP	1725
60000	5	697345	GOT	173274

Table 10 - The execution time of the algorithms on the A1.

ε	MP	OT	τ	10000
			NGP	2530
100000	5	3226123	GOT	540397

Table 11 - The execution time of the algorithms on the S3.

ε	MP	OT	τ	4000
			NGP	378
500000	5	7091022	GOT	1997

Table 12 - The execution time of the algorithms on the Unbalance.

ε	MP	OT	τ	5500
			NGP	3037
600000	5	13664865	GOT	947071

Table 13 - The execution time of the algorithms on the t4.8k.

Figures

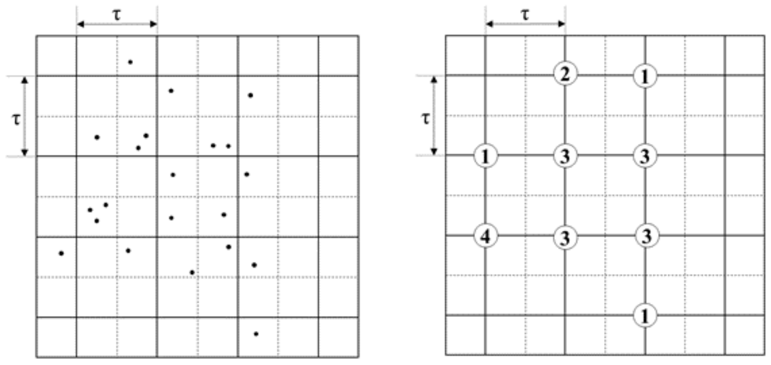

Figure 1: Input points and grid points

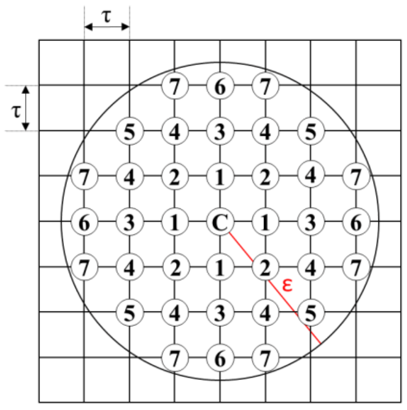

Figure 2 – C point and its neighbours

```
Calculate Core Distances;
while Element Number of Unprocessed Grid Elements != 0
  {
  C is an Element From the Unprocessed Grid Elements
  account C processed
  if C is core-object
    {
    add Neighbours of C to Neighbour Grid Elements
    while Element Number of Neighbour Grid Elements != 0
      {
      calculate Reachability Distances for every
         Element of Neighbour Grid Elements
         with regard to each Processed Element
      S is the Element from Neighbour Grid Elements which
         has the Smallest Reachability Distance
      account S processed
      take out S from Neighbour Grid Elements
      if S is core-object
        add Neighbours of S to Neighbour Grid Elements
      }
    }
  }
```

Figure 3: The pseudo-code of second step - Applying the OPTICS to the grid structure

```
ClusterNumber = 0
ClusterElementNumber = 1
ClusterGridElementNumber =
    Number of Input Points of Processed Grid Elements [0]
Cluster Number of Processed Grid Elements [0] = clusterNumber

for i = 1.. Element Number of Processed Grid Elements
  {
  if Reachability Distance of Processed Grid Elements[i] >= φ
    {
    if (ClusterElementNumber < MinPts)
        for j = 0 .. clusterElementNumber
          Processed Grid Elements [i - 1 - j] is NOISE
      else ClusterNumber++;
    Cluster Number of Processed Grid Elements [i] =
         ClusterNumber
    ClusterElementNumber = 1
    ClusterGridElementNumber
      = Number of Input Points of Processed Grid Elements [i]
    }
  else
    {
    Cluster Number of Processed Grid Elements [i] =
        ClusterNumber
    ClusterElementNumber +=
       Number of Input Points of Processed Grid Elements [i]
    ClusterGridElementNumber++
    }
 }

if (ClusterElementNumber < MinPts)
   for j = 0..ClusterElementNumber
     Processed Grid Elements [Processed Grid
           Elements Number - 1- j] is NOISE
```

Figure 4 - The pseudo-code of the third step - Determining clusters of the grid points

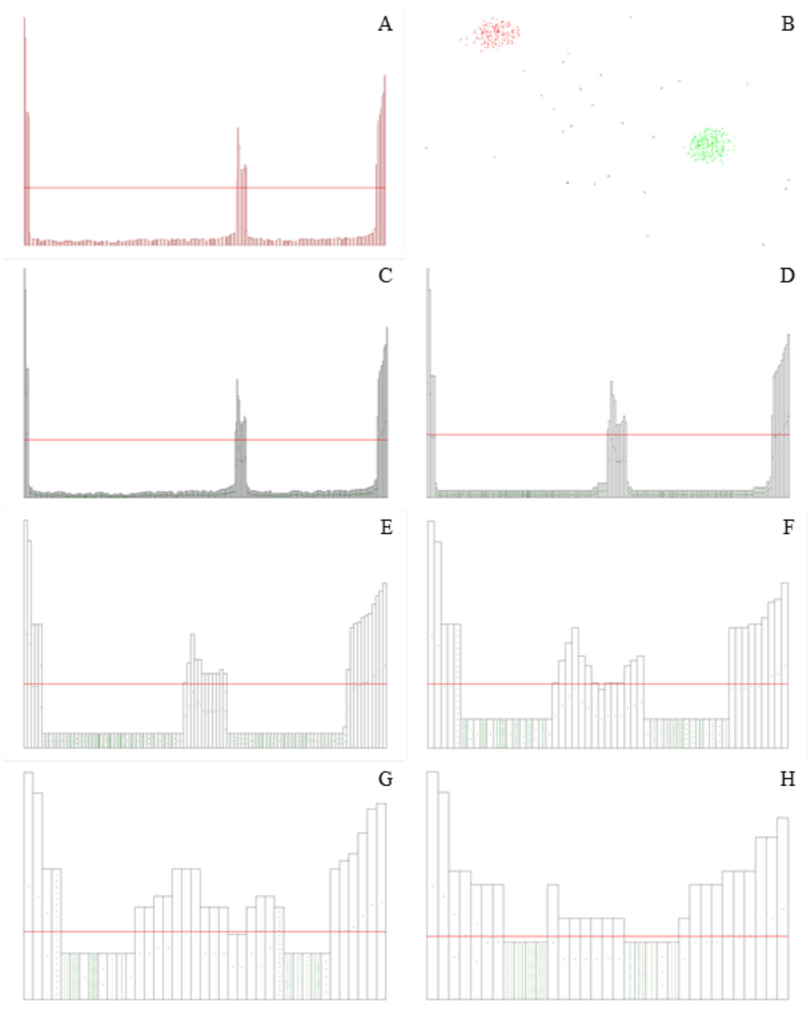

Figure 5 – Results executed on PointSet500 with ε=500, MinPts=5.

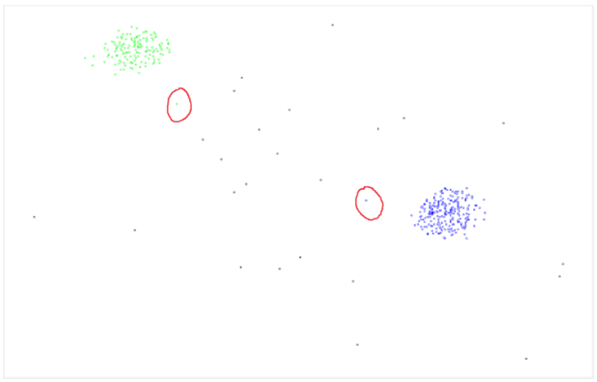

Figure 6 – The clustered points of the PointSet500, executed with τ=40

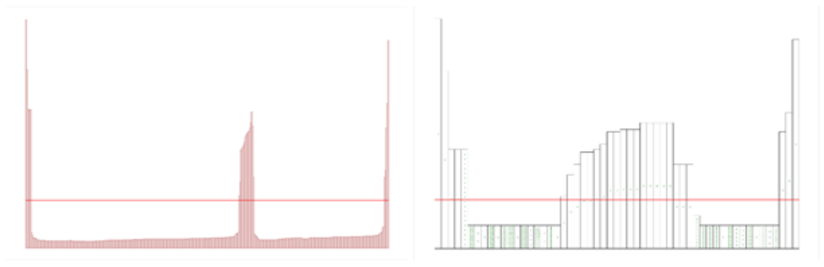
Figure 7 – The reachability plots of the OPTICS (left side) and the GridOPTICS (right side) on the PointSet500 with ε=500, MinPts=20, τ =20, and φ=42.

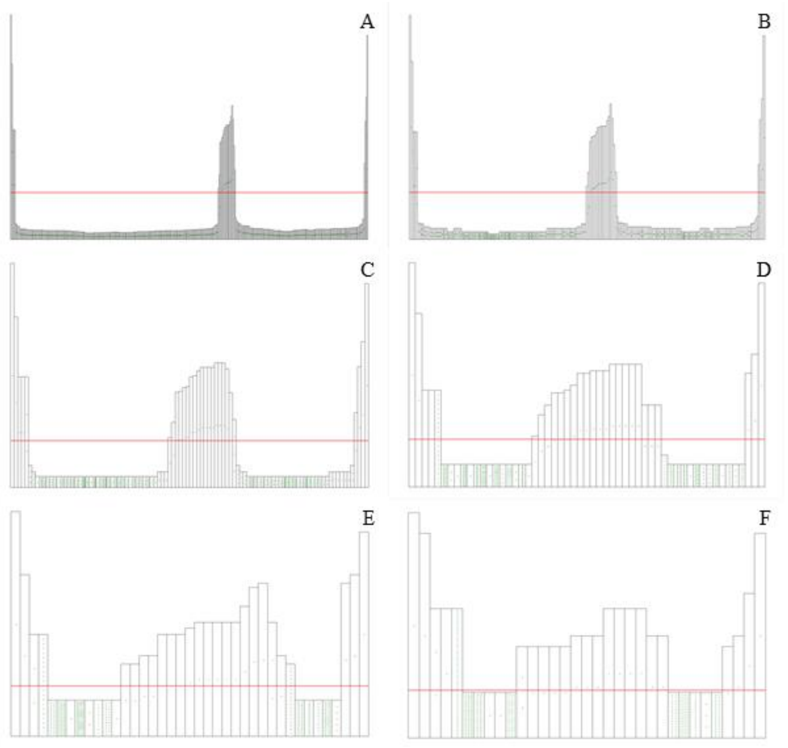

Figure 8 – Results of the GridOPTICS on the PointSet500 with ε=500, MinPts=20, φ=42, τ=1, 5, 10, 20, 30, 40.

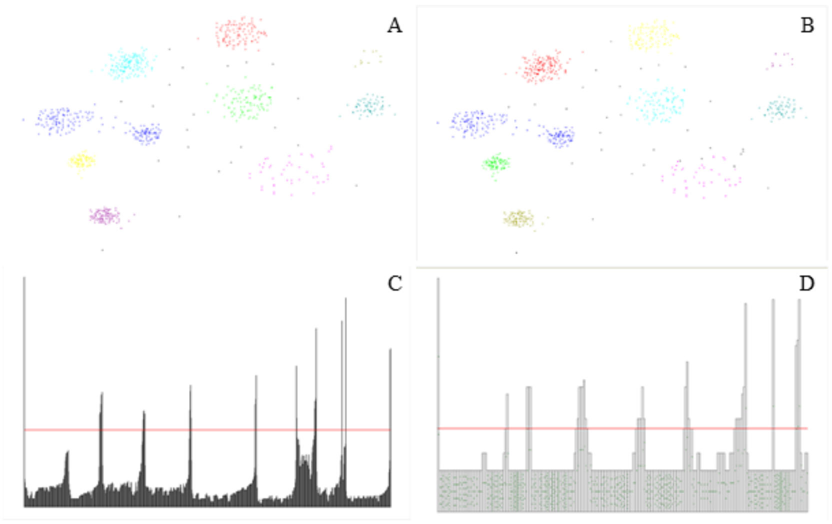

Figure 9 – Results of the OPTICS and the GridOPTICS on the PointSet1000 with ε =400, MinPts=5, τ=10, and φ =20.

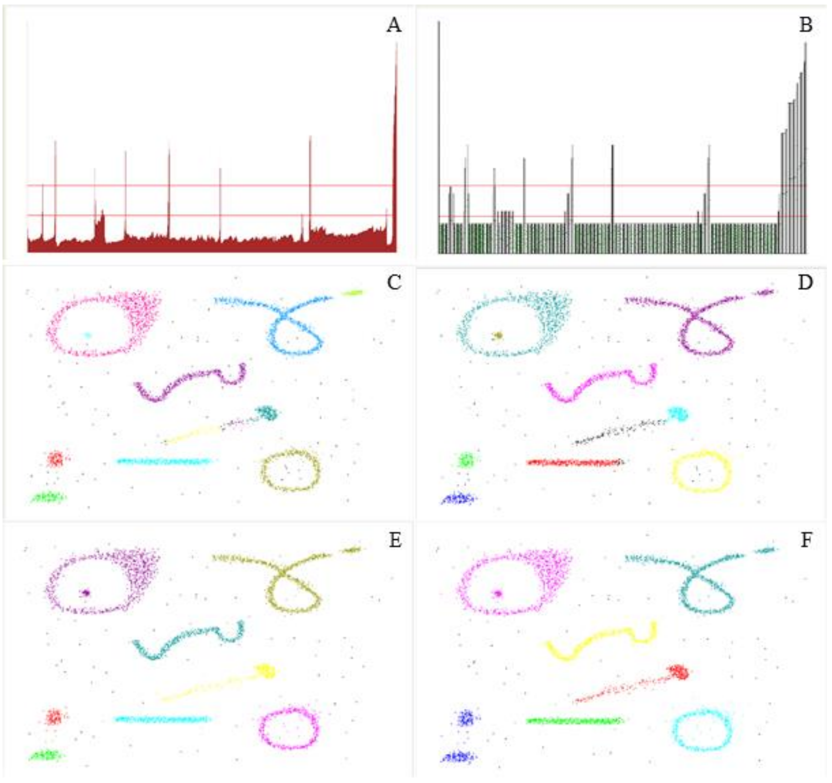

Figure 10 – Results of the OPTICS and the GridOPTICS on the PointSet4000 with ε =1000, MinPts=20, τ=20, φ =25, and φ =45.

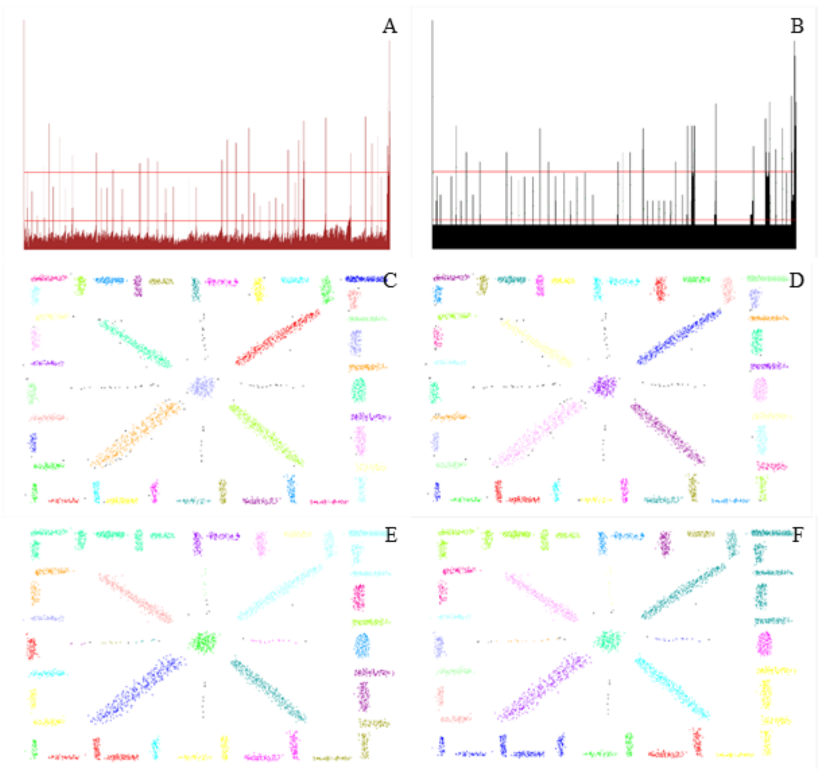

Figure 11 - Results of the OPTICS and the GridOPTICS on the PointSet5000 with ε=1000, MinPts=5, τ =10, φ=12 and φ=32.

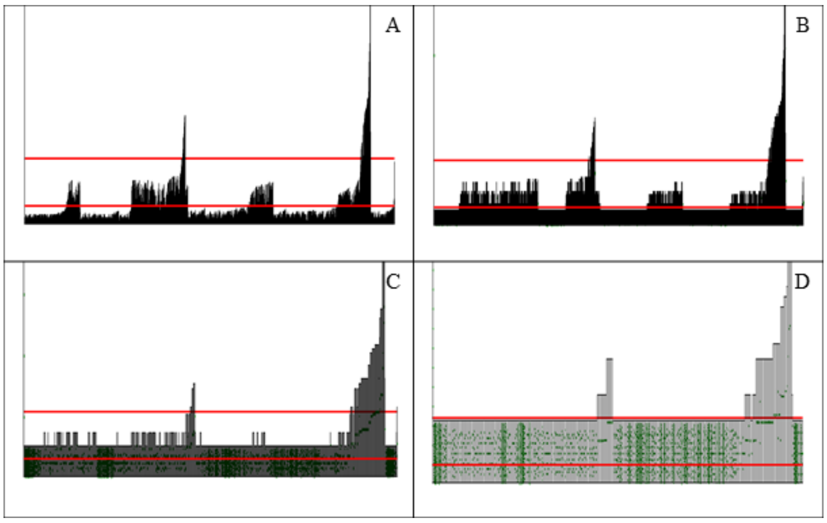

Figure 12 – The reachability plots of the OPTICS and the GridOPTICS on the PointSet3000 with ε=800, MinPts=5, τ =5, 10, 20, φ=6 and φ=21.

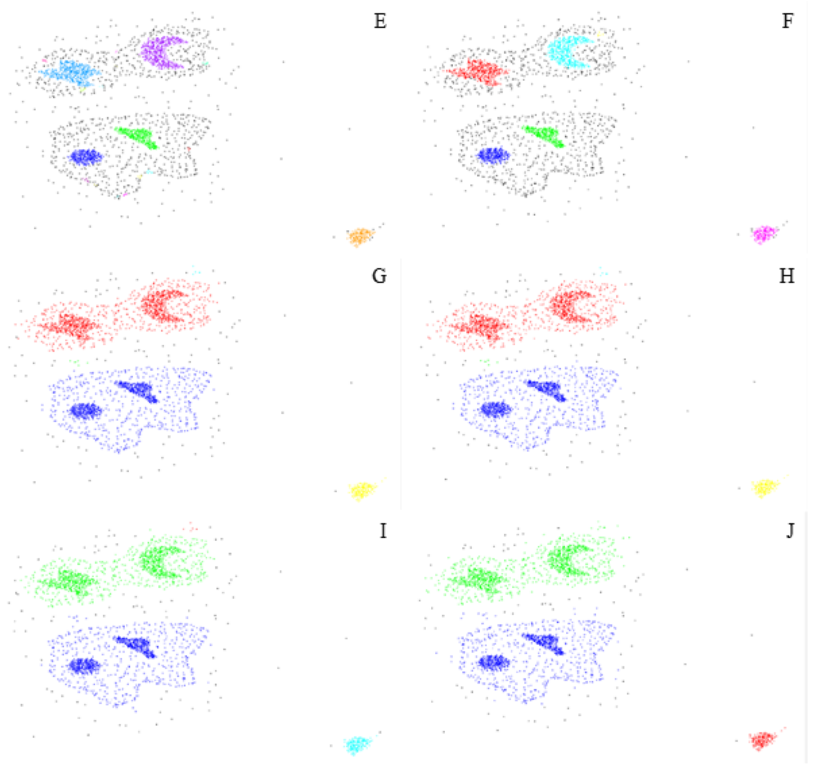

Figure 13 – The clustered points of the OPTICS and the GridOPTICS on the PointSet3000 with ε=800, MinPts=5, τ =5, 10, 20, φ=6 and φ=21.

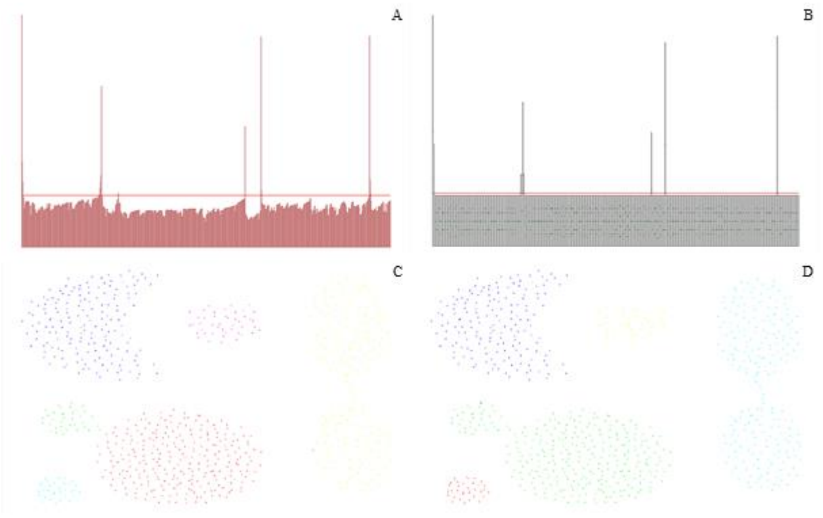

Figure 14 – The reachability plots and clustered points of the OPTICS (A,C) and the GridOPTICS (B,D) on the Aggregation with ε=3000, MinPts=5, τ=110 and φ=115.

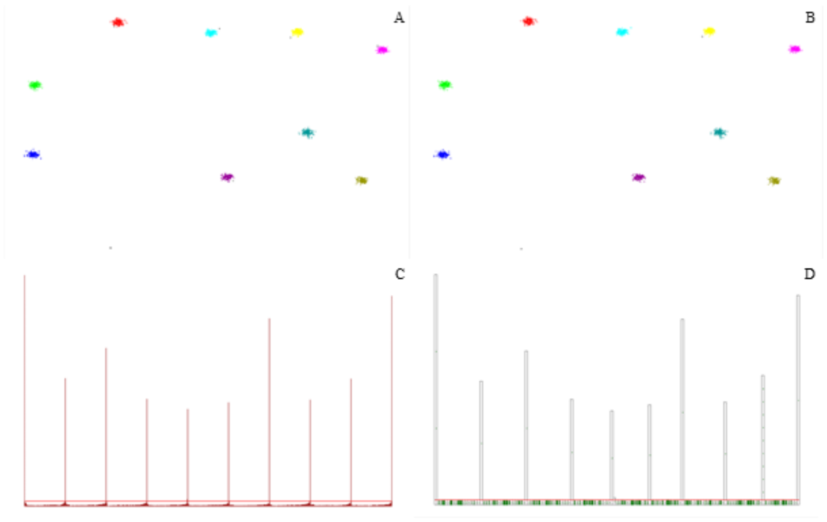

Figure 15 – The reachability plots and clustered points of the OPTICS (A,C) and the GridOPTICS (B,D) on the Dim2 with ε= 1000000, MinPts=5, τ=10000 and φ=10100.

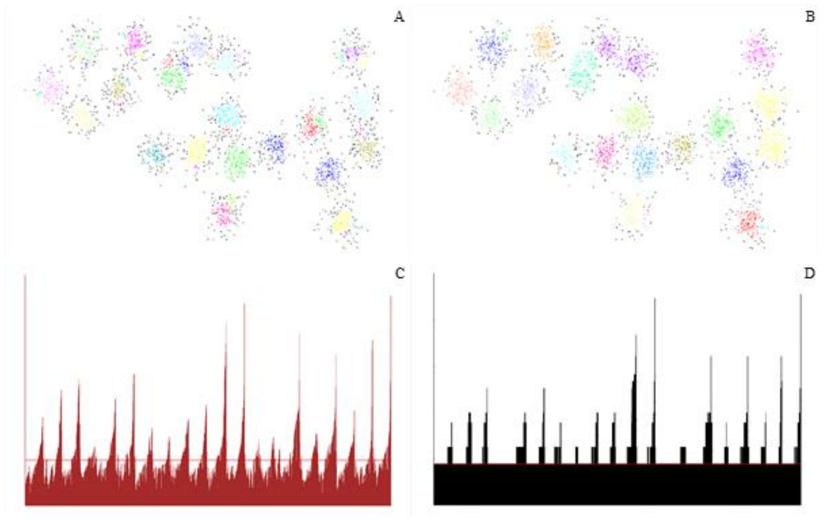

Figure 16 – The reachability plots and clustered points of the OPTICS (A,C) and the GridOPTICS (B,D) on the A1 with ε= 60000, MinPts=5, τ=500 and φ=501.

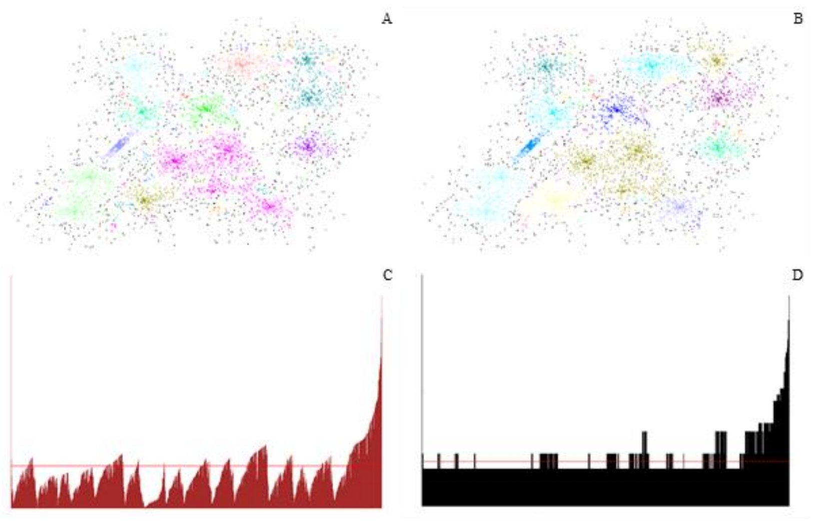

Figure 17 – The reachability plots and clustered points of the OPTICS (A,C) and the GridOPTICS (B,D) on the S3 with ε= 100000, MinPts=5, τ=10000 and φ=12000.

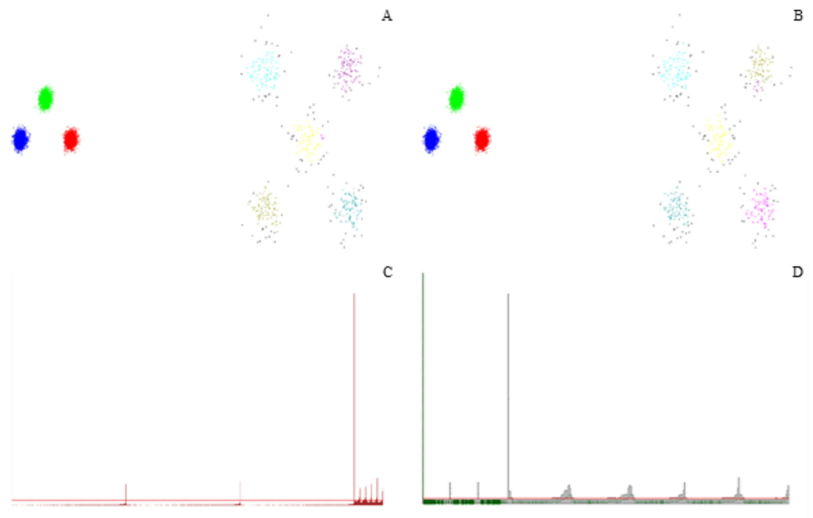

Figure 18 – The reachability plots and clustered points of the OPTICS (A,C) and the GridOPTICS (B,D) on the Unbalance with ε= 500000, MinPts=5, τ=4000 and φ=5000.

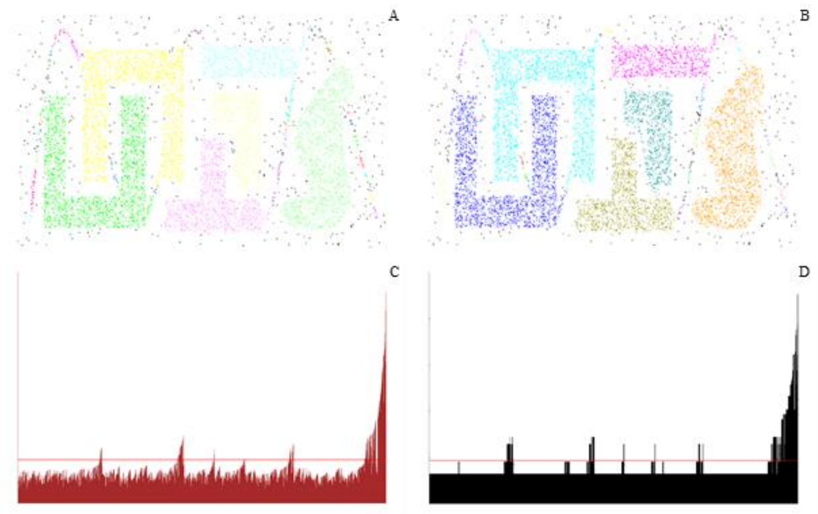

Figure 19 – The reachability plots and clustered points of the OPTICS (A,C) and the GridOPTICS (B,D) on the t4.8k with ε= 600000, MinPts=5, τ= 5500 and φ=5800.

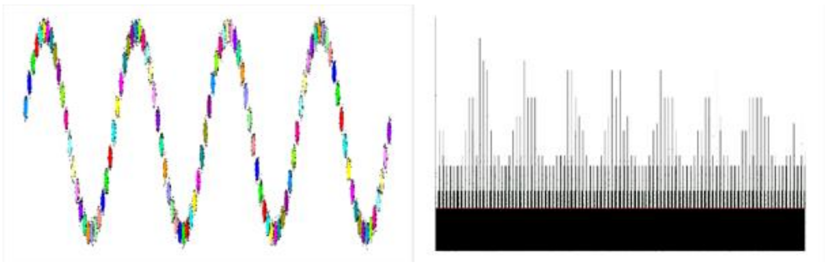

Figure 20 – The clustered points and the reachability plots resulted by the GridOPTICS with ε=10000, MinPts=50, τ =1000, and φ=1010 on BIRCH2 data set (Zhang et al., 1997), whose cardinality is 100000. The execution time is 1279471 milliseconds. The number of the grid points is 7131. The x coordinates range between 47734 and 1000000, whereas the y coordinates range between 0 and 86244.

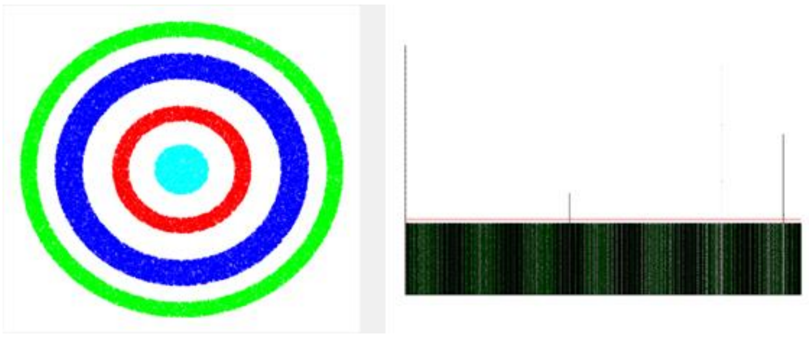

Figure 21 – The clustered points and the reachability plots resulted by the GridOPTICS with ε=800, MinPts=5, τ =200, and φ=21 on PointsetCircle50000 synthetic data set, whose cardinality is 49968. The execution time is 121626 milliseconds. The number of the grid points is

1023. The x coordinates range between 81 and 920, whereas the y coordinates range between 81 and 920.

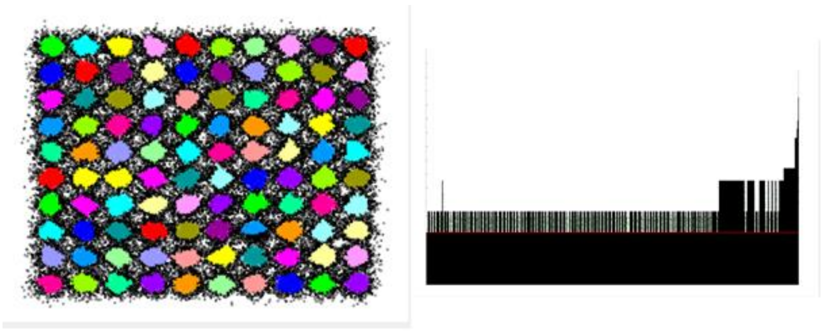
Figure 22 – The clustered points and the reachability plots resulted by the GridOPTICS with ε=100000, MinPts=50, τ =8000, and φ=8010 on BIRCH1 data set (Zhang et al., 1997), whose cardinality is 100000. The execution time is 191413781 milliseconds. The number of the grid points is 13507. The x coordinates range between 1371 and 996108 whereas the y coordinates range between 0 and 1000000.

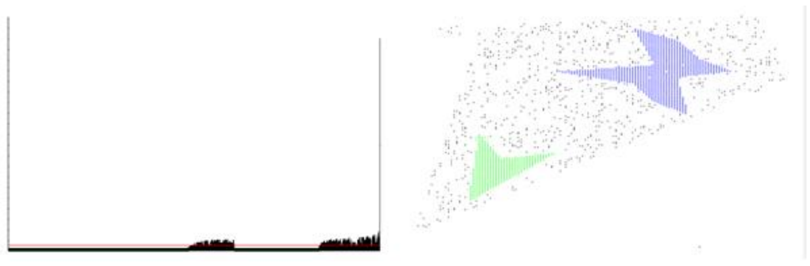
Figure 23 – Results of the GridOPTICS with ε=200, MinPts=5, τ =1, and φ=2 on the PointSet41000, whose cardinality is 41000. The cardinality of the grid points is 2541. The execution time is 1925881 millisecond. The x coordinates range between 27 and 158 whereas the y coordinates range between 14 and 199.

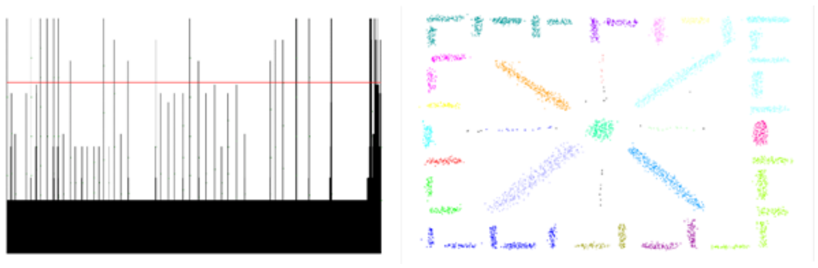
Figure 24 – Results of the GridOPTICS with ε=40, MinPts=5, τ =10 and φ=32 on the PointSet5000.