Clustering Algorithms
- Partitioning Alg: Construct various partitions then evaluate them by some criterion (CLARANS, O(n) calls)
- Hierarchy Alg: Create a hierarchical decomposition of the set of data (or objects) using some criterion (merge & divisive, difficult to find termination condition)
- Density-based Alg: based on local connectivity and density functions

Density-Based Clustering
- Clustering based on density (local cluster criterion), such as density-connected points
- Each cluster has a considerable higher density of points than outside of the cluster

Density Concepts
- Two global parameters:
  - \( \text{Eps} \): Maximum radius of the neighbourhood
  - \( \text{MinPts} \): Minimum number of points in an Eps neighbourhood of that point
- Core Object: object with at least \( \text{MinPts} \) objects within a radius \( \text{Eps} \)-neighborhood
- Border Object: object that on the border of a cluster

\[
\text{MinPts} = 5 \\
\text{Eps} = 1 \text{ cm}
\]
Density-Based Clustering: Background

- \( N_{Eps}(p) \): \( q \in D \mid \text{dist}(p,q) \leq Eps \)
- Directly density-reachable: A point \( p \) is directly density-reachable from a point \( q \) wrt. \( Eps, MinPts \) if
  1) \( p \in N_{Eps}(q) \)
  2) \(|N_{Eps}(q)| \geq MinPts \)
  (core point condition)

Density-Based Clustering: Background (II)

- Density-reachable:
  - A point \( p \) is density-reachable from a point \( q \) wrt. \( Eps, MinPts \) if there is a chain of points \( p_0, \ldots, p_n \) in \( D \) such that \( p_0 = q, p_n = p \) such that \( p_i \) is directly density-reachable from \( p_{i-1} \).

- Density-connected
  - A point \( p \) is density-connected to a point \( q \) wrt. \( Eps, MinPts \) if there is a point \( o \) such that both, \( p \) and \( q \) are density-reachable from \( o \) wrt. \( Eps, MinPts \).

DBSCAN: Density Based Spatial Clustering of Applications with Noise

- Relies on a density-based notion of cluster: A cluster is defined as a maximal set of density-connected points
- Discovers clusters of arbitrary shape in spatial databases with noise

DBSCAN: The Algorithm

- Arbitrary select a point \( p \)
- Retrieve all points density-reachable from \( p \) wrt \( Eps \) and \( MinPts \).
  - If \( p \) is a core point, a cluster is formed.
  - If \( p \) is a border point, no points are density-reachable from \( p \) and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed.

DBSCAN: Core, Border and Noise Points

- Core points
- Border points
- Outliers

When DBSCAN Works Well

- Resistant to Noise
- Can handle clusters of different shapes and sizes
R*-Tree(1)
- R*-Tree: A spatial index
- Generalize the 1-dimensional B+Tree to d-dimensional data spaces

R*-Tree(2)
- R*-Tree is a height-balanced data structure
- Search key is a collection of d-dimensional intervals
- Search key value is referred to as bounding boxes

R*-Tree(3)
- Query a bounding box B in R*-Tree:
  - Test bounding box for each child of root
  - If it overlaps B, search the child’s subtree
  - If more than one child of root has a bounding box overlapping B, we must search all the corresponding subtrees
  - Important difference between B+tree: search for single point can lead to several paths

DBSCAN Complexity Comparison

<table>
<thead>
<tr>
<th>Time Complexity</th>
<th>A single neighborhood query</th>
<th>DBSCAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without index</td>
<td>O(d)</td>
<td>O(n)</td>
</tr>
<tr>
<td>R*-tree</td>
<td>O(log n)</td>
<td>O(n log n)</td>
</tr>
</tbody>
</table>

The height of a R*-Tree is O(log n) in the worst case.
A query with a “small” region traverses only a limited number of paths in the R*-Tree.
For each point, at most one neighborhood query is needed.

Heuristic for Eps and Minpts
- K-dist (p): distance from the kth nearest neighbour to p
- Sorting by k-dist (p)
- Minpts: k>4 no significant difference, but more computation, thus set k=4
Performance Evaluation compared with CLARANS (1)

Accuracy

<table>
<thead>
<tr>
<th>CLARANS:</th>
<th>DBSCAN:</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Clustering Example 1" /></td>
<td><img src="image2" alt="Clustering Example 2" /></td>
</tr>
<tr>
<td><img src="image3" alt="Clustering Example 3" /></td>
<td><img src="image4" alt="Clustering Example 4" /></td>
</tr>
</tbody>
</table>

Performance Evaluation compared with CLARANS (2)

Efficiency

<table>
<thead>
<tr>
<th>SEQUOIA2000 benchmark data (Stonebraker et al. 1993)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>3212</td>
</tr>
<tr>
<td>1026</td>
</tr>
<tr>
<td>4148</td>
</tr>
<tr>
<td>12312</td>
</tr>
</tbody>
</table>

Advantages

- DBSCAN does not require you to know the number of clusters in the data a priori, as opposed to k-means.
- DBSCAN can find arbitrarily shaped clusters. It can even find clusters completely surrounded by (but not connected to) a different cluster. Due to the MinPts parameter, the so-called single-link effect (different clusters being connected by a thin line of points) is reduced.
- DBSCAN has a notion of noise.

Disadvantages

- DBSCAN can only result in a good clustering as good as its distance measure is in the function getNeighbors(P, epsilon). The most common distance metric used is the euclidean distance measure. Especially for high-dimensional data, this distance metric can be rendered almost useless due to the so-called "Curse of dimensionality", rendering it hard to find an appropriate value for epsilon. This effect however is present also in any other algorithm based on the euclidean distance.
- DBSCAN cannot cluster data sets well with large differences in densities, since the minPts-epsilon combination cannot be chosen appropriately for all clusters then.

Advantages

- DBSCAN requires just two parameters and is mostly insensitive to the ordering of the points in the database. (Only points sitting on the edge of two different clusters might swap cluster membership if the ordering of the points is changed, and the cluster assignment is unique only up to isomorphism.)

Disadvantages

- DBSCAN can only result in a good clustering as good as its distance measure is in the function getNeighbors(P, epsilon). The most common distance metric used is the euclidean distance measure. Especially for high-dimensional data, this distance metric can be rendered almost useless due to the so-called "Curse of dimensionality", rendering it hard to find an appropriate value for epsilon. This effect however is present also in any other algorithm based on the euclidean distance.
- DBSCAN cannot cluster data sets well with large differences in densities, since the minPts-epsilon combination cannot be chosen appropriately for all clusters then.

References

- Ester M., Kriegel H.-P., Sander J. and Xu X. 1996. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise" Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD), 226-231.
- Rajee Raghavan, Rajeev, "Database Management systems”, McGraw Hill Companies, Inc.