

*(Paper Presentation)*

# OPTICS-Ordering Points To Identify The Clustering Structure

---

Presenter

Anu Singha

Asiya Naz

Rajesh Piryani

South Asian University



# OUTLINE

---

- Introduction
- Definition (Directly Density Reachable, Density Reachable, Density Connected,
- OPTICS Algorithm
- Example
- Graphical Results



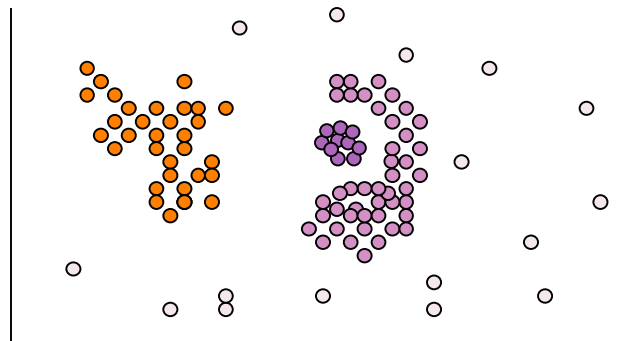
# CLUSTERING

---

- **Goal**
  - Group objects into meaningful subclasses as part of an exploratory process to insight into data or as a preprocessing step for other algorithms.
- **Clustering Strategies**
  - Hierarchical
  - Partitioning
    - k-means
  - **Density Based**

# DENSITY BASED CLUSTERING

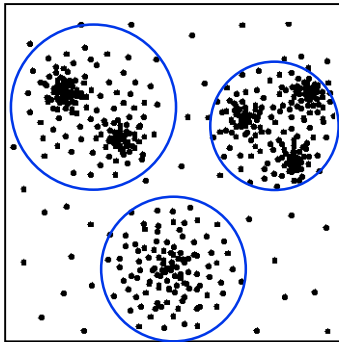
- Density-based Clustering locates regions of high density that are separated from one another by regions of low density.
  - Density = number of points within a specified radius (Eps)



# DENSITY BASED CLUSTERING

## Flat Clustering

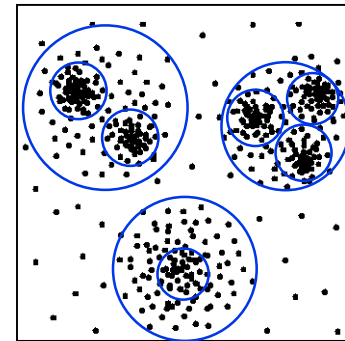
one level of clusters



e.g. density-based clustering algorithm  
DBSCAN [KDD 96]

## Hierarchical Clustering

nested clusters



e.g. density-based clustering algorithm  
OPTICS [SIGMOD 99]



# INTRODUCTION

---

- DBSCAN can cluster objects given input parameters such as
  - (Eps) (the maximum radius of a neighborhood) and
  - MinPts (the minimum number of points required in the neighborhood of a core object),
- it encumbers users with the responsibility of selecting parameter values that will lead to the discovery of acceptable clusters.
- Such parameter settings are usually empirically set and difficult to determine.
- Moreover, real-world, high-dimensional data sets often have very skewed distributions such that their intrinsic clustering structure may not be well characterized by a single set of global density parameters.



# INTRODUCTION

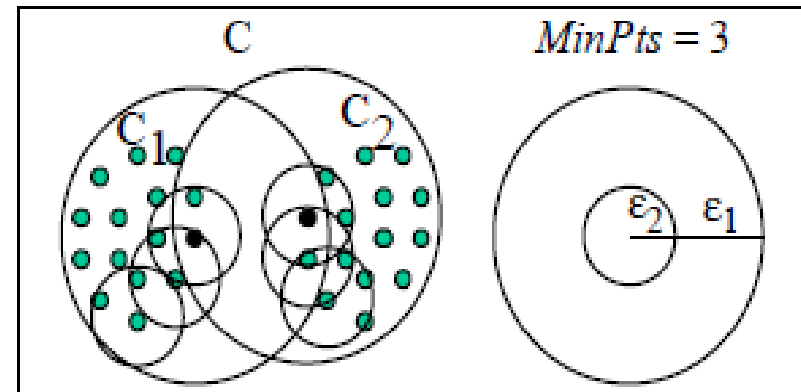
---

- density-based clusters are monotonic with respect to the neighborhood threshold.
- In DBSCAN, for a fixed MinPts value and two neighborhood thresholds,
  - $(Eps)_1 < (Eps)_2$ , a cluster C with respect to  $(Eps)_1$  and
  - MinPts must be a subset of a cluster C' with respect to  $(Eps)_2$  and MinPts.
- This means that if two objects are in a density-based cluster, they must also be in a cluster with a lower density requirement.
- Different clusters may have very different densities
  - Clusters may be in hierarchies

To overcome the difficulty in using one set of global parameters in clustering analysis, a cluster analysis method called **OPTICS was proposed.**

# OPTICS

- in figure 3, where
- $C_1$  and  $C_2$  are density-based clusters with respect to  $\epsilon_2 < \epsilon_1$
- and  $C$  is a density based cluster with respect to  $\epsilon_1$  completely containing the sets  $C_1$  and  $C_2$ .
- for a constant MinPts-value, density-based clusters with respect to a higher density (i.e. a lower value for  $\epsilon$ ) are completely contained in density-connected sets with respect to a lower density (i.e. a higher value for  $\epsilon$ ).



**Figure 3. Illustration of “nested” density-based clusters**





# OPTICS

---

- To produce a consistent result
  - obey a specific order in which objects are processed when expanding a cluster.
- select an object which is density-reachable with respect to the lowest  $\epsilon$  value
  - to guarantee that clusters w.r.t higher density (i.e. smaller  $\epsilon$  values) are finished first.
- OPTICS works in principle like such an extended DBSCAN algorithm for an infinite number of distance parameters  $\epsilon_i$  which are smaller than a “generating distance”  $\epsilon$  (i.e.  $0 \leq \epsilon_i \leq \epsilon$ ).
- The only difference is that we do not assign cluster memberships.
- Instead, we store the order in which the objects are processed and the information which would be used by an extended DBSCAN algorithm to assign cluster memberships if this were at all possible for an infinite number of parameters).



# OPTICS

---

- OPTICS does not explicitly produce a data set clustering.
- It outputs a cluster ordering.
  - It is linear list of all objects under analysis and
  - represents the density-based clustering structure of the data.
- Objects in a denser cluster are listed closer to each other in the cluster ordering.
- Ordering is equivalent to density-based clustering obtained from a wide range of parameter settings.
- Thus, OPTICS does not require the user to provide a specific density threshold.
- The cluster ordering can be used to extract basic clustering information (e.g., cluster centers, or arbitrary-shaped clusters), derive the intrinsic clustering structure, as well as provide a visualization of the clustering



# OPTICS (CONTINUED..)

---

- To construct the different clusterings simultaneously, the objects are processed in a specific order.
- This order selects an object that is density-reachable with respect to the lowest (Eps) value so that clusters with higher density (lower (Eps)) will be finished first.
- Based on this idea, OPTICS needs two important pieces of information per object:
  - Core Distance
  - Reachability Distance

It was presented by Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel and Jörg Sander.

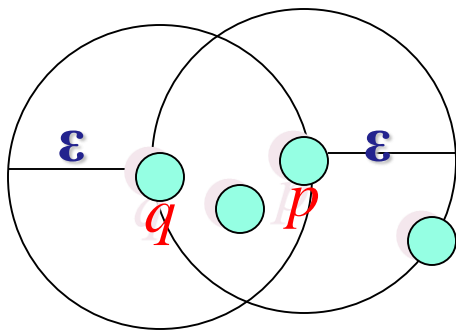
# TERMINOLOGIES

- **$\epsilon$ -Neighborhood**

- Objects within a radius of  $\epsilon$  from an object. (epsilon-neighborhood)

- **Core objects**

- $\epsilon$ -Neighborhood of an object contains at least **MinPts** of objects



$\epsilon$ -Neighborhood of  $p$

$\epsilon$ -Neighborhood of  $q$

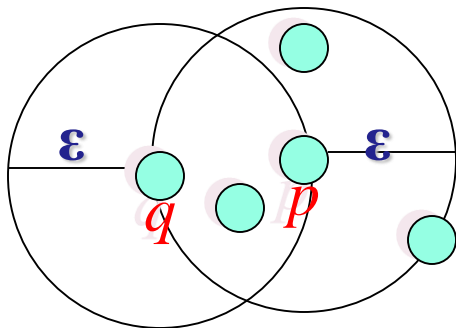
$p$  is a core object ( $\text{MinPts} = 4$ )

$q$  is not a core object

# TERMINOLOGIES

## ■ Directly Density Reachable

- An object  $q$  is directly density-reachable from object  $p$  if  $q$  is within the  $\epsilon$ -Neighborhood of  $p$  and  $p$  is a core object

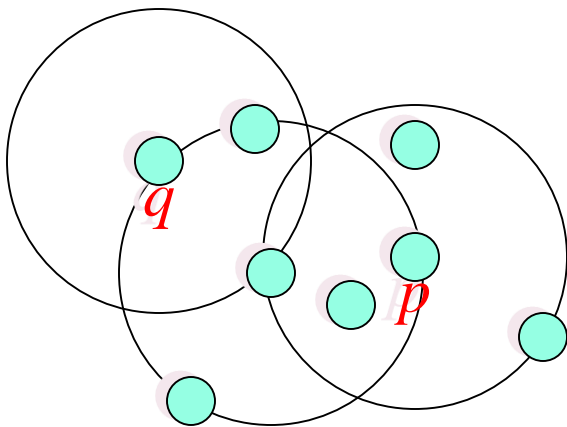


- $q$  is directly density-reachable from  $p$
- $p$  is not directly density-reachable from  $q$

# TERMINOLOGIES

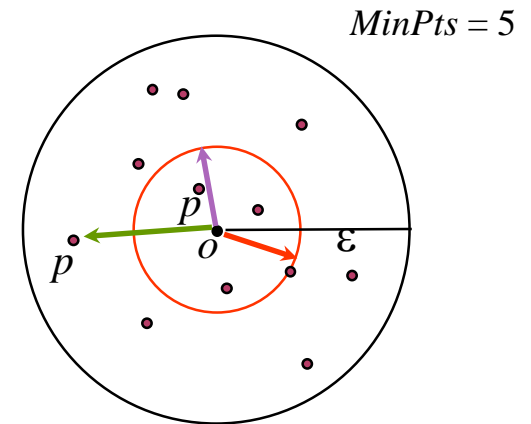
## ■ Density Reachable

- An object  $p$  is density-reachable from  $q$  w.r.t  $\epsilon$  and  $MinPts$  if there is a chain of objects  $p_1, \dots, p_n$ , with  $p_1 = q$ ,  $p_n = p$  such that  $p_{i+1}$  is directly density-reachable from  $p_i$  w.r.t  $\epsilon$  and  $MinPts$  for all  $1 \leq i \leq n$



- $q$  is density-reachable from  $p$
- $p$  is not density-reachable from  $q$
- Transitive closure of direct density-Reachability, asymmetric

# TERMINOLOGIES



## ■ Definition: core-distance

$$\text{core-dist}_{\varepsilon, MinPts}(o) = \begin{cases} \infty & \text{if } |\text{rangeQuery}(o, \varepsilon)| < MinPts \\ MinPts - \text{dist}(o) & \text{otherwise} \end{cases}$$

## ■ Definition: reachability-distance

$$\text{reach-dist}_{\varepsilon, MinPts}(p, o) = \max(\text{core-dist}_{\varepsilon, MinPts}(o), \text{dist}(p, o))$$



# ABOUT OPTICS COMPUTATION

---

- It computes an ordering of all objects in a given database. And
- It stores the **core-distance** and a suitable **reachability-distance** for each object in the database.
- OPTICS maintains a list called OrderSeeds to generate the output ordering.
- Objects in **OrderSeeds**
  - are sorted by the reachability-distance from their respective closest core objects,
  - that is, by the smallest reachability-distance of each object.





# ABOUT OPTICS ALGORITHM

---

- Begin with an **arbitrary object** from the input database as the current object,  $p$ .
- It retrieves the  **$\epsilon$ -neighborhood of  $p$** , **determines the core-distance**, and sets the **reachability-distance to undefined**.
- The current object,  $p$ , is then written to output.
- If  $p$  is not a core object,
  - OPTICS simply moves on to the next object in the *OrderSeeds* list (or the input database if *OrderSeeds* is empty).



# ABOUT OPTICS ALGORITHM

---


- If  $p$  is a core object,
  - then for each object,  $q$ , in the  $\epsilon$ -neighborhood of  $p$ ,
    - OPTICS updates its reachability-distance from  $p$
    - and inserts  $q$  into OrderSeeds if  $q$  has not yet been processed.
- The iteration continues until the input is fully consumed and OrderSeeds is empty.



# ALGORITHM

---

- OPTICS (SetOfObjects, e, MinPts, OrderedFile)
  - OrderedFile.open();
  - FOR i FROM 1 TO SetOfObjects.size DO
    - Object := SetOfObjects.get(i);
    - IF NOT Object.Processed THEN
      - ExpandClusterOrder(SetOfObjects, Object, e, MinPts, OrderedFile)
  - OrderedFile.close();
- END; // OPTICS



# PROCEDURE FOR ExpandClusterOrder

- ExpandClusterOrder(SetOfObjects, Object,  $\epsilon$ , MinPts, OrderedFile);
  - neighbors := SetOfObjects.neighbors(Object,  $\epsilon$ );
  - Object.Processed := TRUE;
  - Object.reachability\_distance := UNDEFINED;
  - Object.setCoreDistance(neighbors,  $\epsilon$ , MinPts);
  - OrderedFile.write(Object);
  - IF Object.core\_distance  $\neq$  UNDEFINED THEN
    - OrderSeeds.update(neighbors, Object);
    - WHILE NOT OrderSeeds.empty() DO
      - currentObject := OrderSeeds.next();
      - neighbors:=SetOfObjects.neighbors(currentObject,  $\epsilon$ );
      - currentObject.Processed := TRUE;
      - currentObject.setCoreDistance(neighbors,  $\epsilon$ , MinPts);
      - OrderedFile.write(currentObject);
      - IF currentObject.core\_distance  $\neq$  UNDEFINED THEN
        - OrderSeeds.update(neighbors, currentObject);
  - END; // ExpandClusterOrder

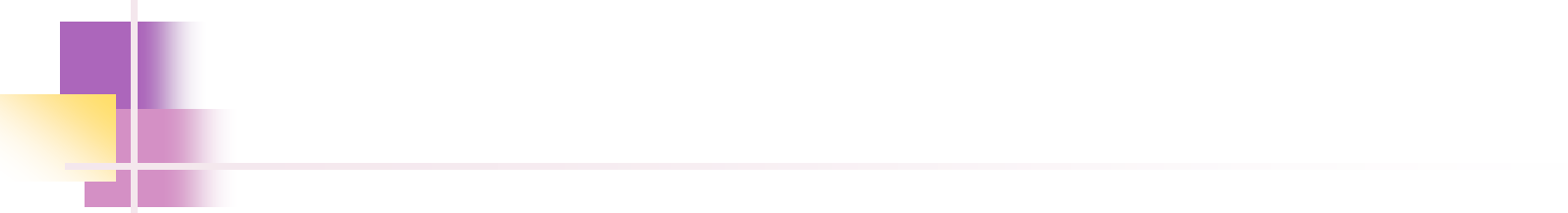
**object is simply written to the file OrderedFile with its coredistance and its current reachability-distance.**



# OrderSeeds::update()

---

- OrderSeeds::update(neighbors, CenterObject);
  - c\_dist := CenterObject.core\_distance;
  - FORALL Object FROM neighbors DO
    - IF NOT Object.Processed THEN
      - new\_r\_dist:=max(c\_dist,CenterObject.dist(Object));
      - IF Object.reachability\_distance=UNDEFINED THEN
        - Object.reachability\_distance := new\_r\_dist;
        - insert(Object, new\_r\_dist);
      - ELSE // Object already in OrderSeeds
        - IF new\_r\_dist<Object.reachability\_distance THEN
          - Object.reachability\_distance := new\_r\_dist;
          - decrease(Object, new\_r\_dist);
  - END; // OrderSeeds::update

- 
- Having generated the augmented cluster-ordering of a database with respect to  $\epsilon$  and MinPts,
  - extract any density-based clustering from this order with respect to MinPts and a clustering- distance  $\epsilon' \leq \epsilon$ 
    - by simply “scanning” the cluster-ordering
    - and assigning cluster-memberships depending on the reachability- distance and the core-distance of the objects.
  - That an extraction is possible only demonstrates that the cluster-ordering of a data set actually contains the information about the intrinsic clustering structure of that data set (up to the generating distance  $\epsilon$ ) .

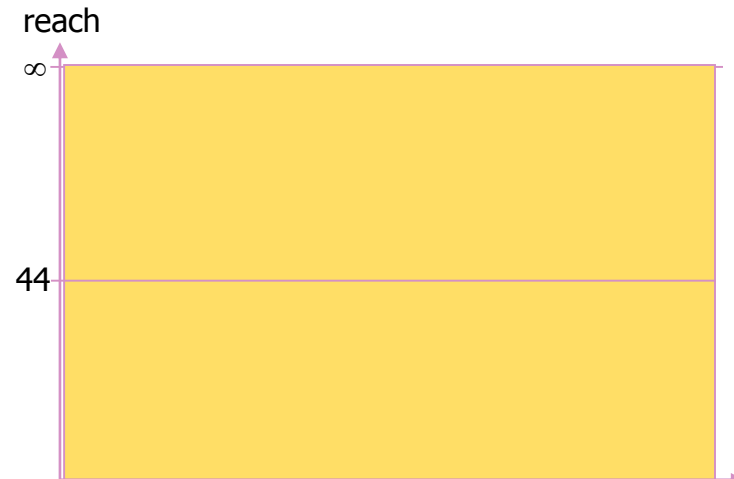
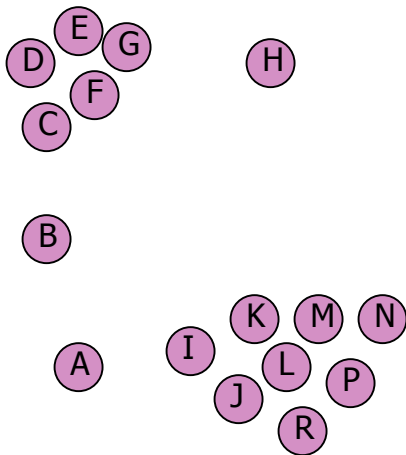


# ExtractDBSCAN-Clustering (ClusterOrderedObjs, $\epsilon'$ , MinPts)

- ExtractDBSCAN-Clustering (ClusterOrderedObjs,  $\epsilon'$ , MinPts)
- // Precondition:  $\epsilon' \leq$  generating dist  $\epsilon$  for ClusterOrderedObjs
  - ClusterId := NOISE;
  - FOR i FROM 1 TO ClusterOrderedObjs.size DO
    - Object := ClusterOrderedObjs.get(i);
    - IF Object.reachability\_distance >  $\epsilon'$  THEN
    - // UNDEFINED >  $\epsilon$ 
      - IF Object.core\_distance  $\leq \epsilon'$  THEN
        - ClusterId := nextId(ClusterId);
        - Object.clusterId := ClusterId;
      - ELSE
        - Object.clusterId := NOISE;
      - ELSE // Object.reachability\_distance  $\leq \epsilon'$ 
        - Object.clusterId := ClusterId;
- END; // ExtractDBSCAN-Clustering

# OPTICS ALGORITHM EXAMPLE

- Example Database (2-dimensional, 16 points)
- $\epsilon = 44$ ,  $MinPts = 3$

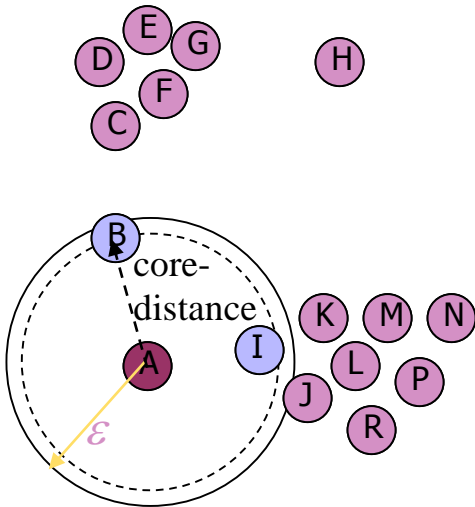


seedlist:



# OPTICS ALGORITHM EXAMPLE

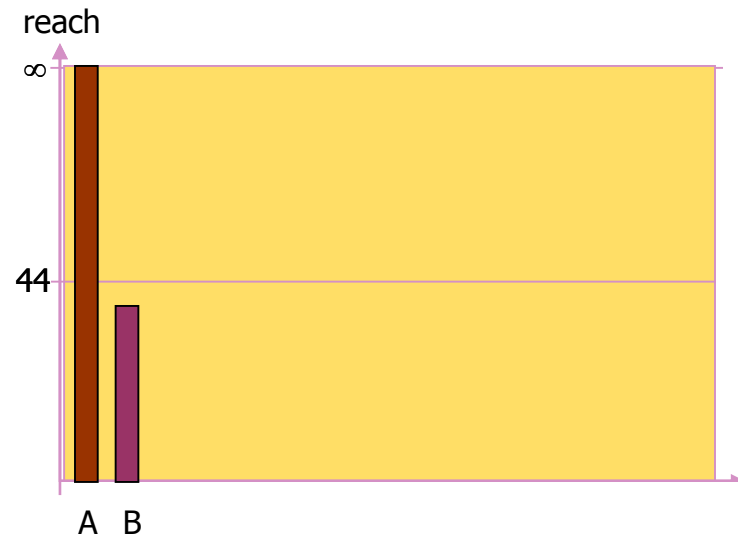
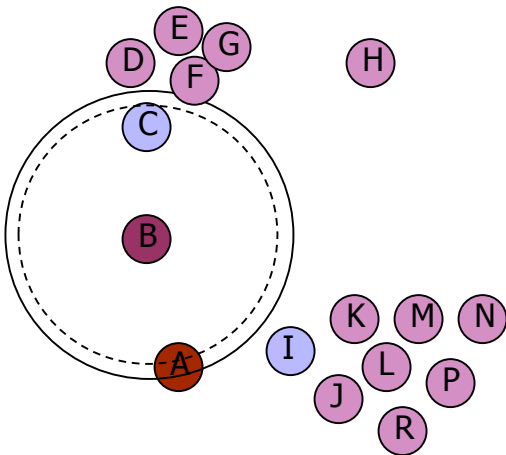
- Example Database (2-dimensional, 16 points)
- $\epsilon = 44$ ,  $MinPts = 3$



seedlist: (B,40) (I, 40)

# OPTICS ALGORITHM EXAMPLE

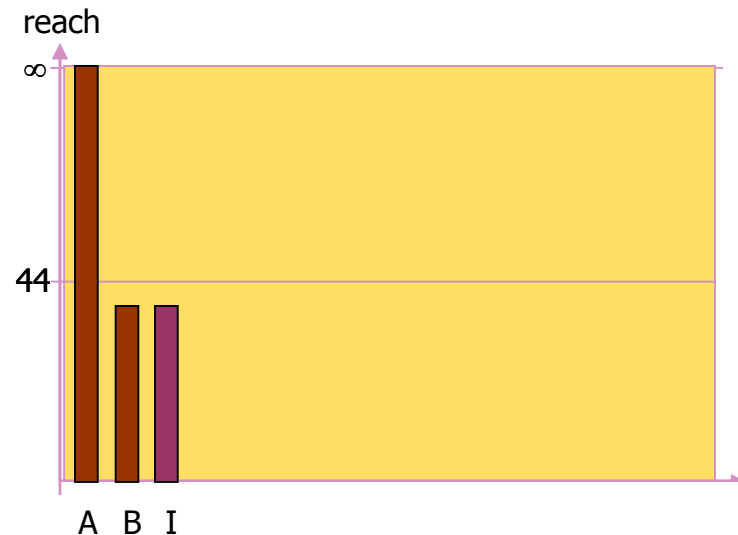
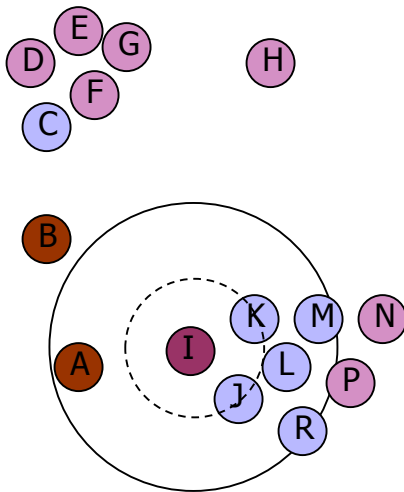
- Example Database (2-dimensional, 16 points)
- $\epsilon = 44$ ,  $MinPts = 3$



seedlist: (I, 40) (C, 40)

# OPTICS ALGORITHM EXAMPLE

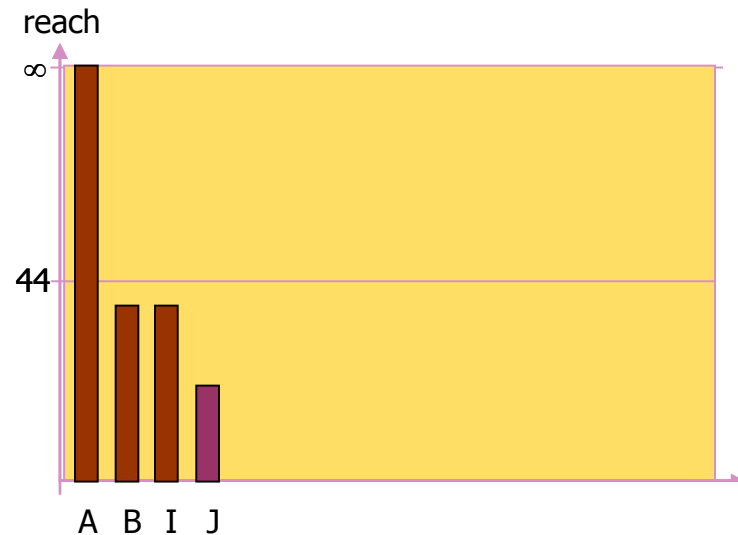
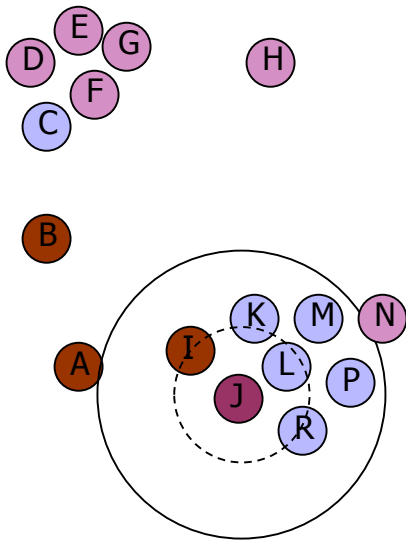
- Example Database (2-dimensional, 16 points)
- $\epsilon = 44$ ,  $MinPts = 3$



seedlist: (J, 20) (K, 20) (L, 31) (C, 40) (M, 40) (R, 43)

# OPTICS ALGORITHM EXAMPLE

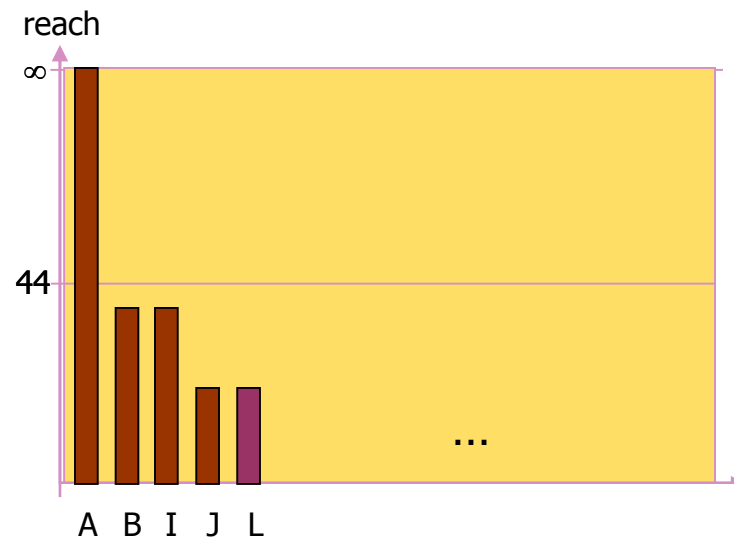
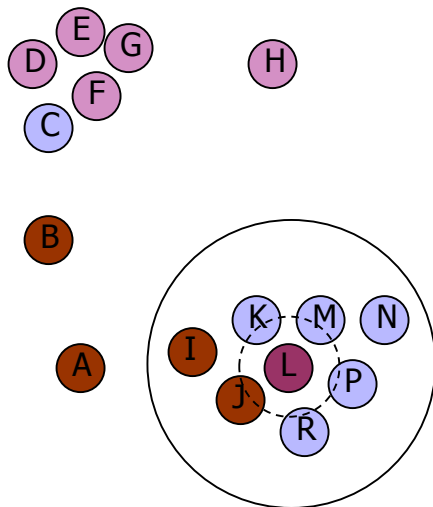
- Example Database (2-dimensional, 16 points)
- $\epsilon = 44$ ,  $MinPts = 3$



seedlist: (L, 19) (K, 20) (R, 21) (M, 30) (P, 31) (C, 40)

# OPTICS ALGORITHM EXAMPLE

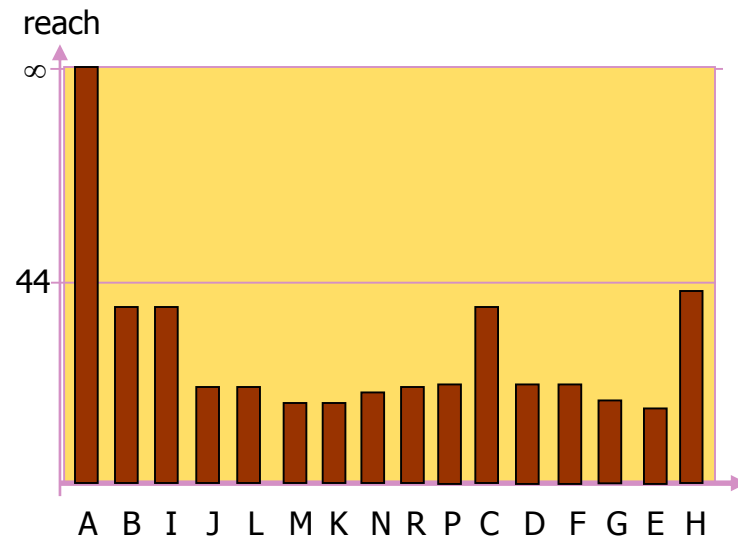
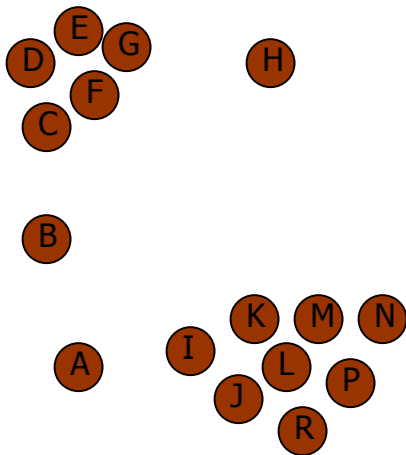
- Example Database (2-dimensional, 16 points)
- $\epsilon = 44$ ,  $MinPts = 3$



seedlist: (M, 18) (K, 18) (R, 20) (P, 21) (N, 35) (C, 40)

# OPTICS ALGORITHM EXAMPLE

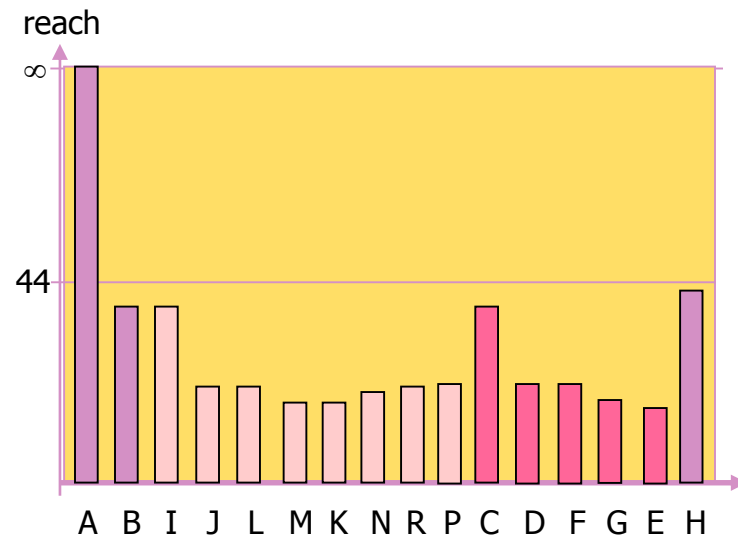
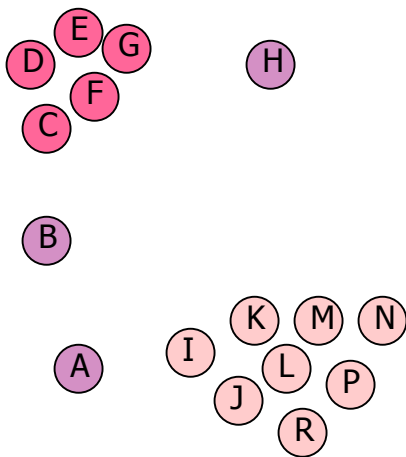
- Example Database (2-dimensional, 16 points)
- $\epsilon = 44$ ,  $MinPts = 3$



seedlist: -

# OPTICS ALGORITHM EXAMPLE

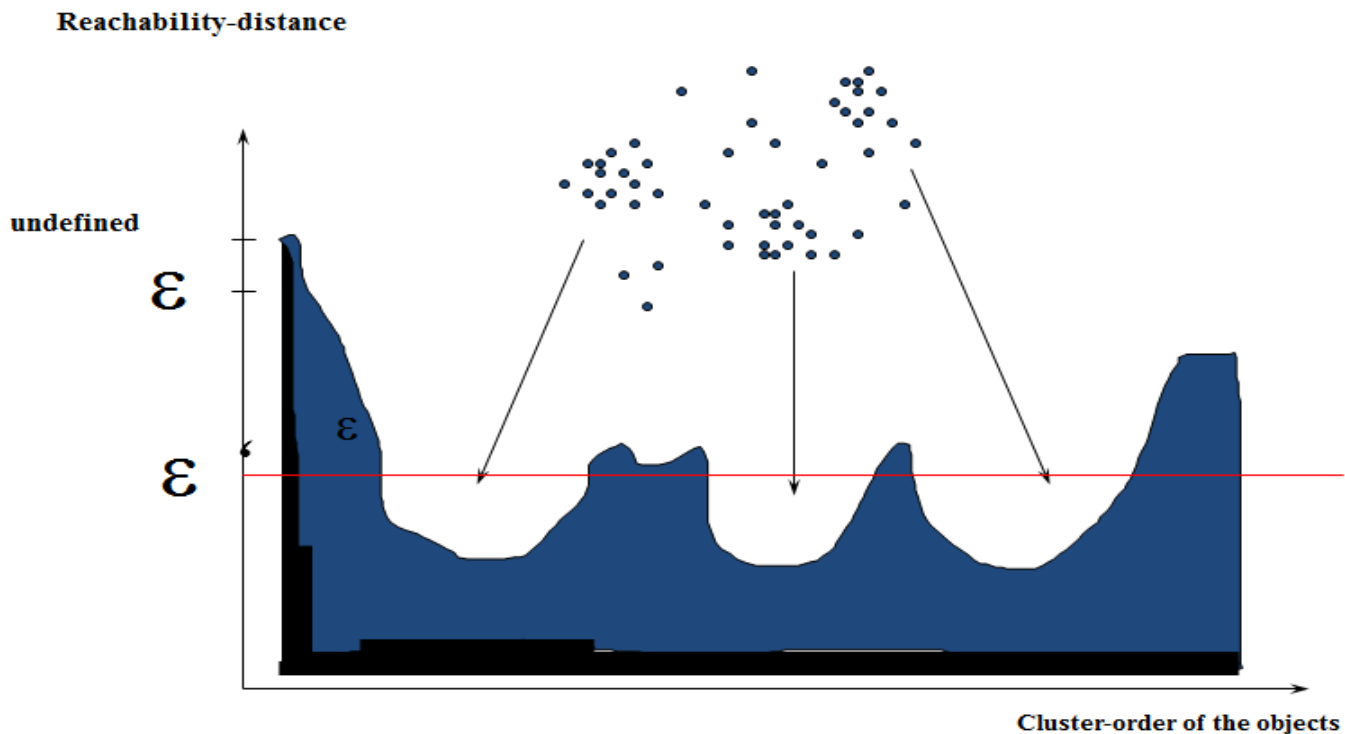
- Example Database (2-dimensional, 16 points)
- $\epsilon = 44$ ,  $MinPts = 3$



seedlist: -

# GRAPHICAL REPRESENTATION

- A data set's cluster ordering can be represented graphically.
- It helps to visualize and understand the clustering structure in a data set.







# GRAPHICAL REPRESENTATION

---

- **In Figure**

- reachability plot for a simple 2-D data set, which presents a general overview of how the data are structured and clustered.
- The data objects are plotted in the clustering order (horizontal axis) together with their respective reachability-distances (vertical axis).
- The three Gaussian “bumps” in the plot reflect three clusters in the data set.



# ALGORITHM PERFORMANCE

---

- performed an extensive performance test using different data sets and different parameter settings.
- simply turned out that the run-time of OPTICS was almost constantly 1.6 times the run-time of DBSCAN.
- not surprising since the run-time for OPTICS as well as for DBSCAN is heavily dominated by
  - the run-time of the  $\epsilon$ -neighborhood queries
  - which must be performed for each object in the database, i.e. the run-time for both algorithms is  $O(n * \text{run-time of an } \epsilon\text{-neighborhood query})$ .



# ALGORITHM PERFORMANCE

---

- To retrieve the  $e$ -neighborhood of an object  $o$ , a region query with the center  $o$  and the radius  $e$  is used.
- Without any index support, to answer such a region query, a scan through the whole database has to be performed.
- In this case, the run-time of OPTICS would be  $O(n^2)$ .
- If a tree-based spatial index can be used, the run-time is reduced to  $O(n \log n)$



# ALGORITHM PERFORMANCE

---

- The height of such a tree-based index is  $O(\log n)$  for a database of  $n$  objects in the worst case and, at least in low-dimensional spaces, a query with a “small” query region has to traverse only a limited number of paths.
- Furthermore, if we have a direct access to the e-neighborhood, e.g. if the objects are organized in a grid, the run-time is further reduced to  $O(n)$  because in a grid the complexity of a single neighborhood query is  $O(1)$ .



# CONCLUSION

---

- OPTICS computes an augmented cluster- ordering of the database objects.
- The main advantage of approach, when compared to the clustering algorithms proposed in the literature, is that, do not limit to one global parameter setting.
- Instead, the augmented cluster-ordering contains information which is equivalent to the density based clusterings corresponding to a broad range of parameter settings and thus is a versatile basis for both automatic and interactive cluster analysis.



# REFERENCES

---

- [1] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, Jörg Sander, “OPTICS: Ordering Points To Identify the Clustering Structure” , Proc. ACM SIGMOD’99 Int. Conf. on Management of Data, Philadelphia PA, 1999.
- [2] Data Mining Concepts and Techniques by Han Kamber Pei , Third Edition
- [3] Stefan Brecheisen, Hans-Peter Kriegel, Martin Pfeifle, “Efficient Density-Based Clustering of Complex Objects“
- [4] Class Lecture Slides about Density Clustering -DBSCAN



THANK YOU  
FOR YOUR CO-OPERATION



# QUESTIONS??