Data Mining and Machine Learning

Bioinspired computational methods

Biological data mining

Cluster Analysis

Francesco Marcelloni

Department of Information Engineering
University of Pisa
ITALY

Some slides belong to the collection

Jiawei Han, Micheline Kamber, and Jian Pei
University of Illinois at Urbana-Champaign
Simon Fraser University

Clustering: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

What is the cluster analysis

- Cluster: A collection of data objects
  - similar (or related) to one another within the same group
  - dissimilar (or unrelated) to the objects in other groups
- Cluster analysis (or clustering, data segmentation, …)
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes (i.e., learning by observations vs. learning by examples: supervised)
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Applications

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Information retrieval: document clustering
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults
- Climate: understanding earth climate, find patterns of atmospheric and ocean
- Economic Science: market research

Clustering as a Preprocessing Tool

- Summarization:
  - Preprocessing for regression, PCA, classification, and association analysis
- Compression:
  - Image processing: vector quantization
- Finding K-nearest Neighbors
  - Localizing search to one or a small number of clusters
- Outlier detection
  - Outliers are often viewed as those "far away" from any cluster

Quality: What is Good Clustering?

- A good clustering method will produce high quality clusters
  - high intra-class similarity: cohesive within clusters
  - low inter-class similarity: distinctive between clusters
- The quality of a clustering method depends on
  - the similarity measure used by the method
  - its implementation, and
  - its ability to discover some or all of the hidden patterns

Measure the Quality of Clustering

- Dissimilarity/Similarity metric
  - Similarity is expressed in terms of a distance function, typically metric: *d(i, j)*
  - The definitions of distance functions are usually rather different for interval-scaled, boolean, categorical, ordinal ratio, and vector variables
  - Weights should be associated with different variables based on applications and data semantics
- Quality of clustering:
  - There is usually a separate "quality" function that measures the "goodness" of a cluster.
  - It is hard to define "similar enough" or "good enough"
    - The answer is typically highly subjective

Considerations for Cluster Analysis

- Partitioning criteria
  - Single level vs. hierarchical partitioning (often, multi-level hierarchical partitioning is desirable)
- Separation of clusters
  - Exclusive (e.g., one customer belongs to only one region) vs. non-exclusive (e.g., one document may belong to more than one class)
- Similarity measure
  - Distance-based (e.g., Euclidian, road network, vector) vs. connectivity-based (e.g., density or contiguity)
- Clustering space
  - Full space (often when low dimensional) vs. subspaces (often in high-dimensional clustering)

Requirements and Challenges

- Scalability
  - Clustering all the data instead of only on samples
- Ability to deal with different types of attributes
  - Numerical, binary, categorical, ordinal, linked, and mixture of these
- Constraint-based clustering
  - User may give inputs on constraints
  - Use domain knowledge to determine input parameters
- Interpretability and usability
- Others
  - Discovery of clusters with arbitrary shape
  - Ability to deal with noisy data
  - Incremental clustering and insensitivity to input order
  - High dimensionality

Major Clustering Approaches

- Partitioning approach:
  - Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
  - Given the number k of partitions to construct, the partitioning method uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another
  - Typical methods: k-means, k-medoids, CLARANS
- Hierarchical approach:
  - Create a hierarchical decomposition of the set of data (or objects) using some criterion
  - Agglomerative versus divisive approaches
  - Suffer from the fact that once a step (merge or split) is done, it can never be undone
  - Typical methods: Diana, Agnes, BIRCH, CHAMELEON

Major Clustering Approaches

- Density-based approach:
  - Based on connectivity and density functions
  - Typical methods: DBSCAN, OPTICS, DenClue
- Grid-based approach:
  - based on a multiple-level granularity structure
  - All the clustering operations are performed on the grid structure (i.e., on the quantized space)
  - Fast processing time
  - Typical methods: STING, WaveCluster, CLIQUE

Major Clustering Approaches

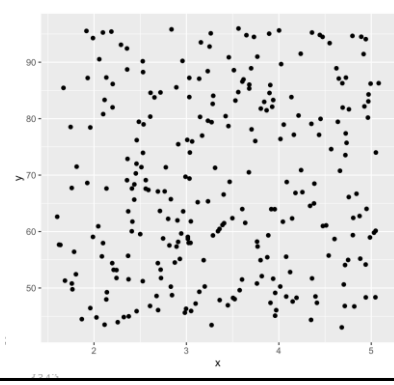
Method	General Characteristics
Partitioning methods	– Find mutually exclusive clusters of spherical shape – Distance-based – May use mean or medoid (etc.) to represent cluster center – Effective for small- to medium-size data sets
Hierarchical methods	– Clustering is a hierarchical decomposition (i.e., multiple levels) – Cannot correct erroneous merges or splits – May incorporate other techniques like microclustering or consider object "linkages"
Density-based methods	– Can find arbitrarily shaped clusters – Clusters are dense regions of objects in space that are separated by low-density regions – Cluster density: Each point must have a minimum number of points within its "neighborhood" – May filter out outliers
Grid-based methods	– Use a multiresolution grid data structure – Fast processing time (typically independent of the number of data objects, yet dependent on grid size)

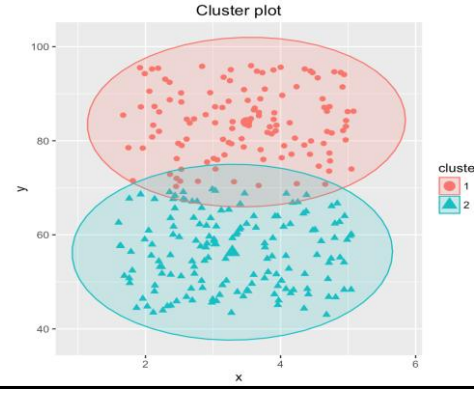
Major Clustering Approaches

- Model-based:
  - A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
  - Typical methods: EM, SOM, COBWEB
- Frequent pattern-based:
  - Based on the analysis of frequent patterns
  - Typical methods: p-Cluster
- User-guided or constraint-based:
  - Clustering by considering user-specified or application-specific constraints
  - Typical methods: COD (obstacles), constrained clustering
- Link-based clustering:
  - Objects are often linked together in various ways
  - Massive links can be used to cluster objects: SimRank, LinkClus

Assessing Clustering Tendency

- Before applying any clustering method on your data, it's important to evaluate whether the data sets contains meaningful clusters (i.e.: non-random structures) or not





Assessing Clustering Tendency

- Clustering tendency assessment determines whether a given data set has a non-random structure, which may lead to meaningful clusters.
  - Uniformly distributed points in a data space
- How can we assess the clustering tendency of a dataset?
- Hopkins statistic: spatial statistic that tests the spatial randomness of a variable as distributed in a space. It is used to assess the clustering tendency of a data set by measuring the probability that a given data set is generated by a uniform data distribution. In other words, it tests the spatial randomness of the data.
  - Given a dataset D which is regarded as a sample of a random variable X, we want to determine how far away X is from being uniformly distributed in the data space.

Assessing Clustering Tendency Hopkins statistic

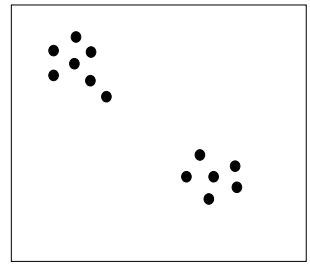
1. Sample n<<N points p_1, …, p_n, uniformly from D. For each point p_i, find its nearest neighbor in D. Let x_i the distance between p_i and its nearest neighbor in D

$$x_i = \min_{v \in D} \{dist(p_i, v)\}$$

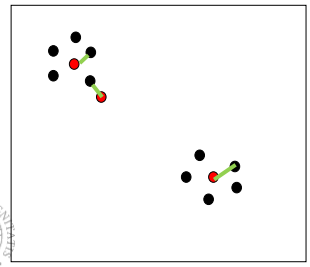
2. Generate a simulated data set ($random_D$) drawn from a random uniform distribution with *n* points q_1, …, q_n and the same variation as the original dataset D. For each point q_i, find the nearest neighbor of q_i in D. Let y_i the distance between q_i and its nearest neighbor in D

$$y_i = \min_{v \in D, v \neq q_i} \{dist(q_i, v)\}.$$

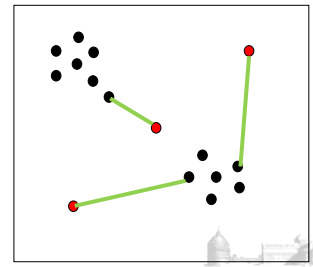
Assessing Clustering Tendency
Hopkins statistic



Randomly from D



Uniform distribution



Assessing Clustering Tendency
Hopkins statistic

3. Calculate the Hopkins Statistic

$$H = \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i}$$

- If D were uniformly distributed, then the two terms at the denominator would be close to each other and therefore H would be about 0.5.
- In presence of clustering tendency, then the first term at the denominator would be smaller than the second and then H will increase.

Assessing Clustering Tendency Hopkins statistic

- The null and alternative hypotheses are defined as follows:
  - Null hypothesis: the data set D is uniformly distributed (i.e., no meaningful clusters)
  - Alternative hypothesis: the data set D is not uniformly distributed (i.e., contains meaningful clusters)
- The Hopkins statistic is computed for several random selection of points and the average of all results for H is used for a decision: if H is larger than 0.75, it indicates a clustering tendency at the 90% confidence level.

Clustering: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

Partitioning Algorithms: Basic Concept

- Partitioning method: Partitioning a database D of n objects into a set of k clusters, such that the sum of squared distances is minimized (where c_i is the centroid or medoid of cluster C_i)

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} (p - c_i)^2$$

- Given k, find a partition of k clusters that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions!!
  - Heuristic methods: k-means and k-medoids algorithms
  - k-means (MacQueen'67, Lloyd'57/'82): Each cluster is represented by the center of the cluster
  - k-medoids or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects in the cluster

The k-means clustering method

The centroid is defined as the mean value of the points within the cluster

Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- k: the number of clusters,
- D: a data set containing n objects.

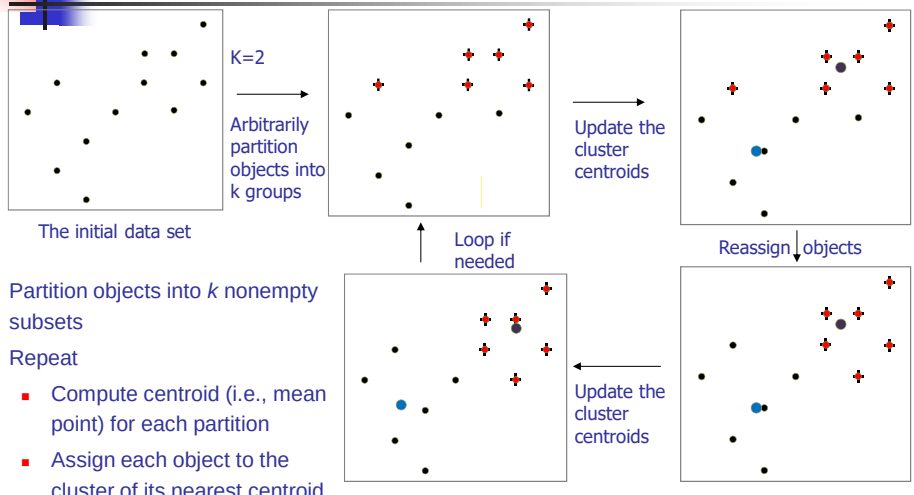
Output: A set of k clusters.

$$c_i = \frac{1}{|C_i|} \sum_{p \in C_i} p$$

Method:

(1) arbitrarily choose k objects from D as the initial cluster centers;
(2) **repeat**
(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) update the cluster means, that is, calculate the mean value of the objects for each cluster;
(5) **until** no change;

An Example of K-means Clustering



- Partition objects into *k* nonempty subsets
- Repeat
  - Compute centroid (i.e., mean point) for each partition
  - Assign each object to the cluster of its nearest centroid
- Until no change

Comments on the K-means Method

- Strength
  - *Efficient*: *O*(tkn), where *n* is # objects, *k* is # clusters, and *t* is # iterations. Normally, *k*, *t* << *n*.
    - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$, where s is the size of the data sample
    - Comment: Often terminates at a local optimal.
- Weakness
  - Applicable only to objects in a continuous n-dimensional space
    - Using the k-modes method for categorical data
    - In comparison, k-medoids can be applied to a wide range of data
  - **Need to specify k**, the number of clusters, in advance (there are ways to automatically determine the best k)
  - **Sensitive to noisy data and outliers**
  - Not suitable for discovering clusters with **non-convex shapes**

Determining the Number of Clusters

- Elbow method
  - increasing the number of clusters can help to reduce the sum of within-cluster variance of each cluster.
  - the marginal effect of reducing the sum of within-cluster variances may drop if too many clusters are formed, because splitting a cohesive cluster into two gives only a small reduction.
  - Use the turning point in the curve of sum of within cluster variance w.r.t the # of clusters

Determining the Number of Clusters

- Elbow method

Technically, given a number, k > 0,

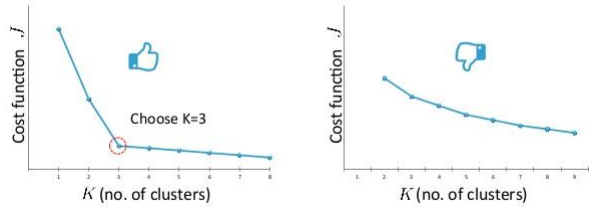
- Form k clusters on the data set in question using a clustering algorithm like k-means, and
- Calculate the sum of within-cluster variances, var(k).
- Plot the curve of var with respect to k.
- Choose *k* as the first or most significant turning point of the curve

Determining the Number of Clusters

- Elbow method

Choosing the value of K

Elbow method:



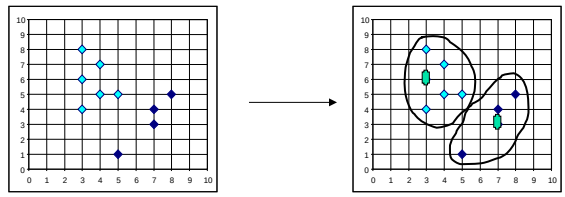
27

Variations on the *K-means* Method

- Most of the variants of the k-means differ in:
  - Selection of the initial k means
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- Handling categorical data: k-modes
  - Replacing means of clusters with modes (values that occur most frequently in a dataset)
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters
  - A mixture of categorical and numerical data: k-prototype method

What Is the Problem of the K-Means Method?

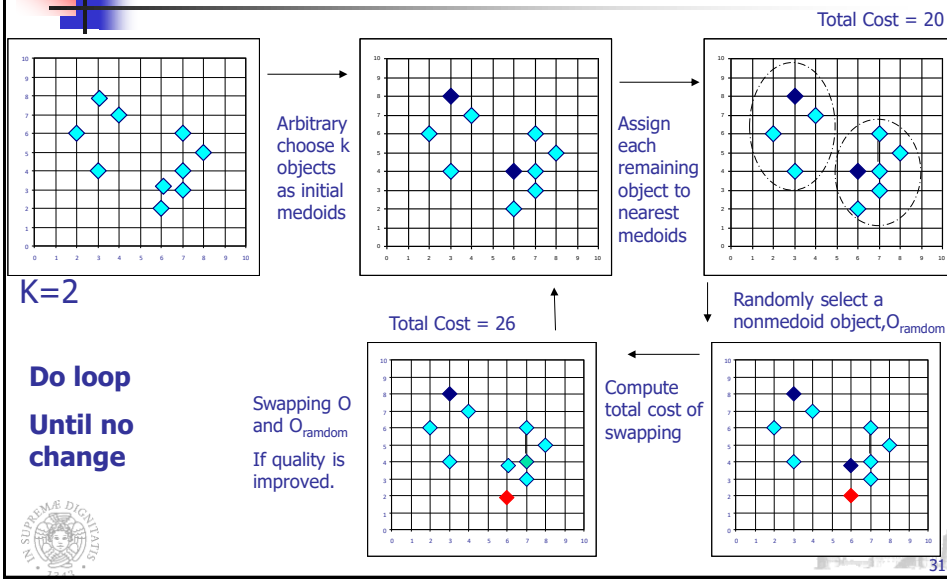
- The k-means algorithm is sensitive to outliers!
  - Since an object with an extremely large value may substantially distort the distribution of the data
- K-Medoids: Instead of taking the mean value of the object in a cluster as a reference point, medoids can be used
- Medoid: the most centrally located object in a cluster



K-Medoids Method: The idea

- Initial representatives are chosen randomly
- The iterative process of replacing representative objects by no representative objects continues as long as the quality of the clustering is improved
- For each representative Object O
  - For each non-representative object R, swap O and R
- Choose the configuration with the lowest cost
- Cost function is the difference in absolute error-value if a current representative object is replaced by a non-representative object

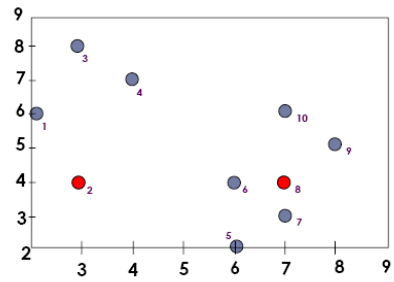
PAM: A Typical K-Medoids Algorithm



K-Medoids Method: Example

Data Objects

	A_1	A_2
O_1	2	6
O_2	3	4
O_3	3	8
O_4	4	7
O_5	6	2
O_6	6	4
O_7	7	3
O_8	7	4
O_9	8	5
O_{10}	7	6



Goal: create two clusters

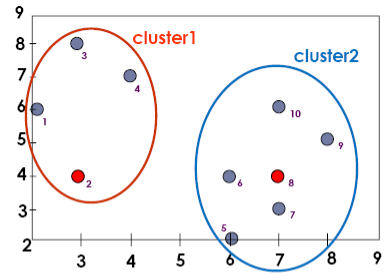
Choose randomly two medoids

$O_2 = (3,4)$
$O_8 = (7,4)$

K-Medoids Method: Example

Data Objects

	A_1	A_2
O_1	2	6
O_2	3	4
O_3	3	8
O_4	4	7
O_5	6	2
O_6	6	4
O_7	7	3
O_8	7	4
O_9	8	5
O_{10}	7	6



→Assign each object to the closest representative object

→Using L1 Metric (Manhattan), we form the following clusters

Cluster1 = {O_1, O_2, O_3, O_4}

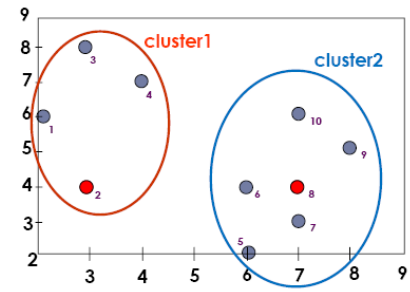
Cluster2 = {O_5, O_6, O_7, O_8, O_9, O_{10}}

33

K-Medoids Method: Example

Data Objects

	A_1	A_2
O_1	2	6
O_2	3	4
O_3	3	8
O_4	4	7
O_5	6	2
O_6	6	4
O_7	7	3
O_8	7	4
O_9	8	5
O_{10}	7	6



→Compute the absolute error criterion **[for the set of Medoids (O2,O8)]**

$$E=\sum_{i=1}^{k}\sum_{p\in C_i}|p-o_i| = |o_1-o_2|+|o_3-o_2|+|o_4-o_2|$$

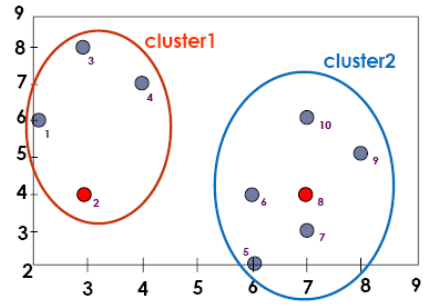
$$+|o_5-o_8|+|o_6-o_8|+|o_7-o_8|+|o_9-o_8|+|o_{10}-o_8|$$

34

K-Medoids Method: Example

Data Objects

	A_1	A_2
O_1	2	6
O_2	3	4
O_3	3	8
O_4	4	7
O_5	6	2
O_6	6	4
O_7	7	3
O_8	7	4
O_9	8	5
O_{10}	7	6



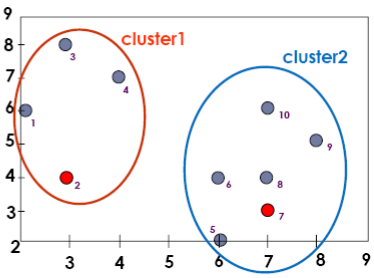
→The absolute error criterion **[for the set of Medoids (O2,O8)]**

$$E = (3+4+4)+(3+1+1+2+2) = 20$$

K-Medoids Method: Example

Data Objects

	A_1	A_2
O_1	2	6
O_2	3	4
O_3	3	8
O_4	4	7
O_5	6	2
O_6	6	4
O_7	7	3
O_8	7	4
O_9	8	5
O_{10}	7	6



→Compute the cost function

Absolute error [for O_2,O_7] – Absolute error [O_2,O_8]

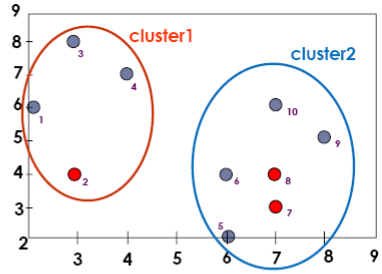
$$S = 22 - 20$$

S> 0 ⇒ it is a bad idea to replace O_8 by O_7

K-Medoids Method: Example

Data Objects

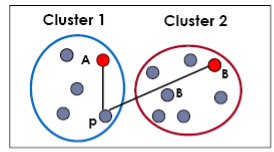
	A_1	A_2
O_1	2	6
O_2	3	4
O_3	3	8
O_4	4	7
O_5	6	2
O_6	6	4
O_7	7	3
O_8	7	4
O_9	8	5
O_{10}	7	6



- In this example, changing the medoid of cluster 2 did not change the assignments of objects to clusters.
- What are the possible cases when we replace a medoid by another object?

37

K-Medoids Method: Example

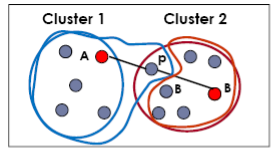


- Representative object
- Random Object

Currently **P** assigned to **A**

First case

The assignment of **P** to **A** does **not change**



- Representative object
- Random Object

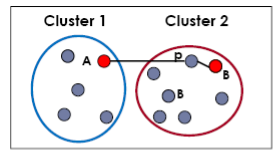
Currently **P** assigned to **B**

Second case

P is **reassigned** to **A**

38

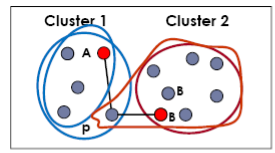
K-Medoids Method: Example



Representative object
Random Object
Currently P assigned to B

Third case

P is reassigned to the new B



Representative object
Random Object
Currently P assigned to A

Fourth case

P is reassigned to B

39

The K-Medoid Clustering Method

PAM (Partitioning Around Medoids)

- **Input**
  - → K: the number of clusters
  - → D: a data set containing n objects
- **Output**: A set of k clusters
- **Method**:

(1) Arbitrary choose k objects from D as representative objects (seeds)
(2) **Repeat**
(3) Assign each remaining object to the cluster with the nearest representative object
(4) For each representative object O_j
(5) Randomly select a non representative object O_{random}
(6) Compute the total cost **S** of swapping representative object Oj with O_{random}
(7) if S<0 then replace O_j with O_{random}
(8) **Until** no change

40

The original PAM Algorithm

PAM (Partitioning Around Medoids) Kaufmann & Rousseeuw 1987

The actual algorithm of pam proceeds in two steps:

1. Step BUILD
   Construct initial 'medoids':
   - m_1 is the object with the smallest $\sum_{i=1}^{n} d(i, m_1)$
   - m_2 decreases the objective obj as much as possible
   - ⋮
   - m_k decreases the objective obj as much as possible

$$\text{obj} = \sum_{i=1}^{n} \min_{t=1,\ldots,k} d(i, m_t)$$

The original PAM Algorithm

PAM (Partitioning Around Medoids) Kaufmann & Rousseeuw 1987

2. Step SWAP
   Repeat until convergence:
   Consider all pairs of objects (i, j) with

$$i \in \{m_1, \ldots, m_k\} \quad \text{and} \quad j \notin \{m_1, \ldots, m_k\}$$

and make the $i \leftrightarrow j$ swap (if any) which decreases the objective most.

- Actually, PAM depends on only a dissimilarity matrix between objects
- The original PAM is computationally very heavy

What Is the Problem with PAM?

- Pam is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean
- Pam works efficiently for small data sets but does not scale well for large data sets.
  - $O(k(n-k))^2$ for each iteration
    where n is # of data, k is # of clusters
- Efficiency improvement on PAM

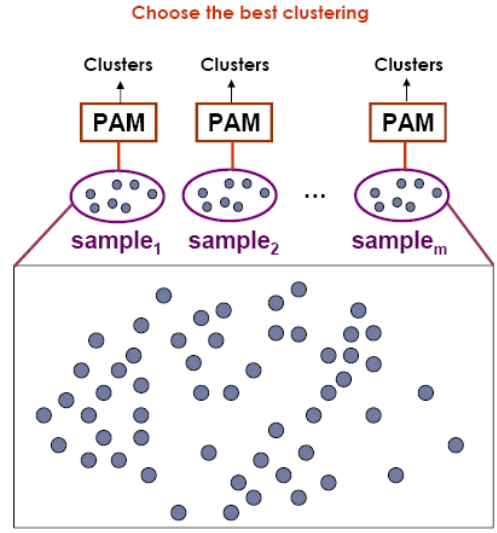
CLARA (Clustering Large Applications) (1990)

CLARA (Kaufmann and Rousseeuw in 1990)

- Draw a sample of the dataset and applies PAM on the sample in order to find the medoids
- If the sample is representative the medoids of the sample should approximate the medoids of the entire dataset.
- Medoids are chosen from the sample. Note that the algorithm cannot find the best solution if one of the best k-medoids is not among the selected sample.
- To improve the approximation, multiple samples are drawn and the best clustering is returned as the output
- The clustering accuracy is measured by the average dissimilarity of all objects in the entire dataset.

CLARA (Clustering Large Applications) (1990)

- CLARA (Kaufmann and Rousseeuw in 1990)
  - It draws multiple samples of the data set, applies PAM on each sample, and gives the best clustering as the output



CLARA (Clustering Large Applications) (1990)

For i= 1 to R, repeat the following steps

a) Draw a sample of objects randomly from the entire data set, and call the algorithm PAM to find k medoids of the sample.

b) For each object in the entire data set, determine which of the k medoids is the most similar to it.

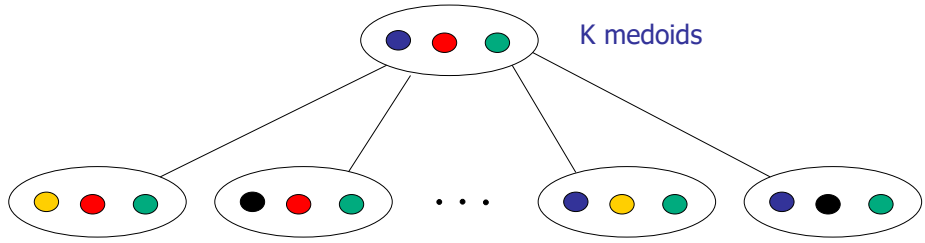
c) Calculate the average dissimilarity ON THE ENTIRE DATASET of the clustering obtained in the previous step. If this value is less than the current minimum, use this value as the current minimum, and retain the k medoids found in Step (b) as the best set of medoids obtained so far.

CLARA (Clustering Large Applications) (1990)

- CLARA (Kaufmann and Rousseeuw in 1990)
- Strength: deals with larger data sets than PAM
  - Complexity $O(ks^2 + k(n-k))$, where s is the size of the sample, k the number of clusters and n the number of objects
- Weakness:
  - Efficiency depends on the sample size
  - A good clustering based on samples will not necessarily represent a good clustering of the whole data set if the sample is biased

CLARANS ("Randomized" CLARA) *(2002)*

- CLARANS (A Clustering Algorithm based on Randomized Search) (Ng and Han 2002)
  - The clustering process can be presented as searching a graph where every node is a potential solution, that is, a set of k medoids
  - Two nodes are neighbours if their sets differ by only one medoid



CLARANS ("Randomized" CLARA) *(2002)*

- CLARANS (A Clustering Algorithm based on Randomized Search) (Ng and Han 2002)
  - Each node is associated with a cost that is defined to be the total dissimilarity between every object and the medoid of its cluster
  - The problem corresponds to search for a minimum on the graph
  - At each step, all neighbours of current_node node are searched; the neighbour which corresponds to the deepest descent in cost is chosen as the next solution

CLARANS ("Randomized" CLARA) *(2002)*

- CLARANS (A Clustering Algorithm based on Randomized Search) (Ng and Han 2002)
  - For large values of n and k, examining k(n-k) neighbours is time consuming.
  - At each step, CLARANS draws sample of neighbours to examine.
  - Note that CLARA draws a sample of nodes at the beginning of search; therefore, CLARANS has the benefit of not confining the search to a restricted area.
  - If the local optimum is found, CLARANS starts with a new randomly selected node in search for a new local optimum. The number of local optimums to search for is a parameter.
  - It is more efficient and scalable than both PAM and CLARA; returns higher quality clusters. Complexity is O(n)

CLARANS ("Randomized" CLARA) *(2002)*

Algorithm CLARANS

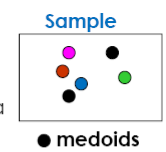
1. Input parameters *numlocal* and *maxneighbor*. Initialize i to 1, and *mincost* to a large number.
2. Set *current* to an arbitrary node in $G_{n,k}$.
3. Set j to 1.
4. Consider a random neighbor S of *current*, and based on 5, calculate the cost differential of the two nodes.
5. If S has a lower cost, set *current* to S, and go to Step 3.
6. Otherwise, increment j by 1. If $j \leq maxneighbor$, go to Step 4.
7. Otherwise, when $j > maxneighbor$, compare the cost of *current* with *mincost*. If the former is less than *mincost*, set *mincost* to the cost of *current* and set *bestnode* to *current*.
8. Increment i by 1. If $i > numlocal$, output *bestnode* and halt. Otherwise, go to Step 2.

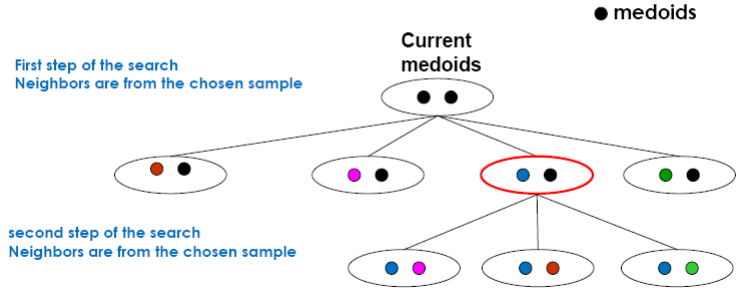
Each node in $G_{n,k}$ is represented by a set of k objects

Two nodes are neighbors (i.e., connected by an arc) if their sets differ by only one object. More formally, two nodes $S_1=\{O_{m1},\ldots, O_{mk}\}$ and $S_2 = \{O_{w1},\ldots, O_{wk}\}$ are neighbors if and only if the cardinality of the intersection of S_1, S_2 is k-1.

CLARA

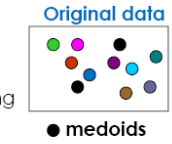
CLARA
- Draws a sample of nodes at the beginning of the search
- Neighbors are from the chosen sample
- Restricts the search to a specific area of the original data

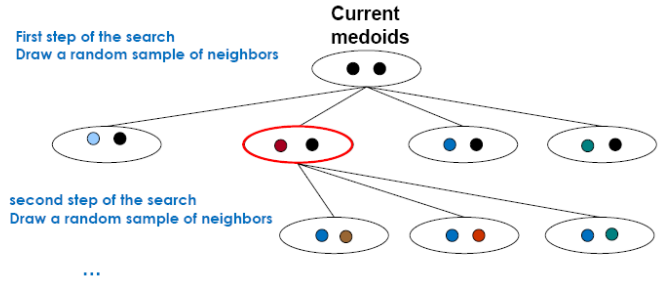




CLARANS

CLARANS
- Does not confine the search to a localized area
- Stops the search when a local minimum is found
- Finds several local optimums and output the clustering with the best local optimum





The number of neighbors sampled from the original data is specified by the user

The K-Medoid Clustering Method

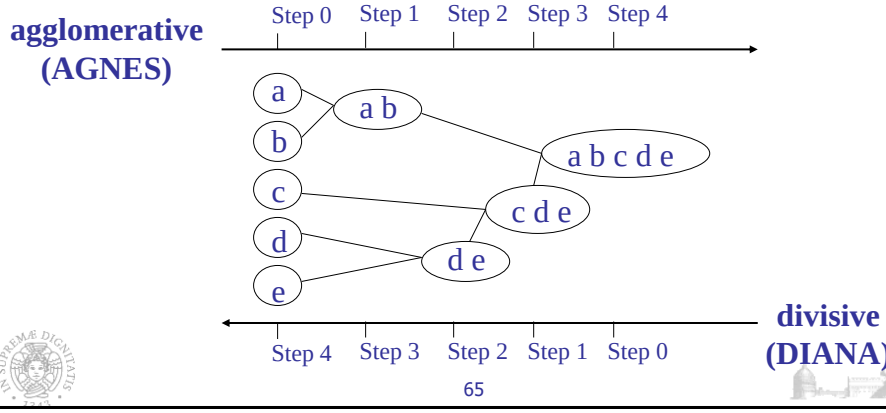
- In conclusion, the clustering process can be presented as searching a graph where every node is a potential solution, that is, a set of *k* medoids. Two nodes are neighbors in the graph if their sets differ by only one object.
  - PAM examines all the neighbors of the current node in its search for a minimum cost
  - CLARA draws a sample of nodes at the beginning of a search
  - CLARANS dynamically draws a random sample of neighbors in each step of a search.

Clustering: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

Hierarchical clustering

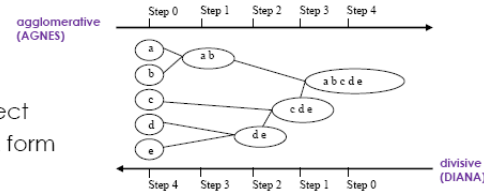
- Use distance matrix as clustering criteria. This method does not require the number of clusters k as an input, but needs a termination condition



Hierarchical clustering

- **AGNES**

→ Clusters C1 and C2 may be merged if an object in C1 and an object in C2 form the minimum Euclidean distance between any two objects from different clusters



- **DIANA**

→ A cluster is split according to some principle, e.g., the maximum Euclidian distance between the closest neighboring objects in the cluster

Hierarchical clustering

- Hierarchical clustering frequently deals with the matrix of distances (dissimilarities) or similarities between training samples.
- It is sometimes called connectivity matrix.
- To merge or split subsets of points rather than individual points, the distance between individual points has to be generalized to the distance between subsets. Such derived proximity measure is called a linkage metric. Linkage metrics are constructed from elements of the connectivity matrix.
- The type of the linkage metric used significantly affects hierarchical algorithms, since it reflects the particular concept of closeness and connectivity.

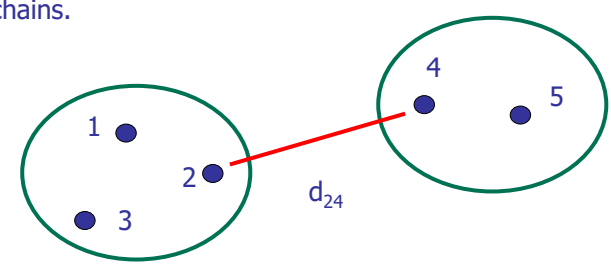
Hierarchical clustering

- Major inter-cluster linkage metrics include single link, average link, and complete link.
- The underlying dissimilarity measure (usually, distance) is computed for every pair of points with one point in the first set and another point in the second set.
- A specific operation such as minimum (single link), average (average link), or maximum (complete link) is applied to pair-wise dissimilarity measures:

$$d(C_1, C_2) = operation\{d(x, y) \mid x \in C_1, y \in C_2\}$$

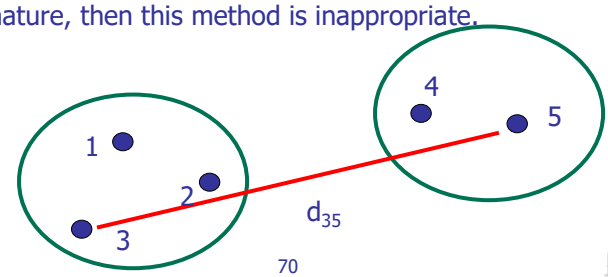
Hierarchical clustering

- Single link (nearest neighbor). The distance between two clusters is determined by the distance of the two closest objects (nearest neighbors) in the different clusters.
  - This rule will, in a sense, string objects together to form clusters, and the resulting clusters tend to represent long chains.



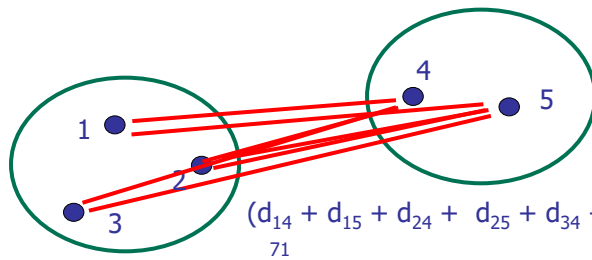
Hierarchical clustering

- Complete link (furthest neighbor). The distance between two clusters is determined by the greatest distance between any two objects (furthest neighbors) in the different clusters.
  - This method usually performs quite well in cases when the objects actually form naturally distinct "clumps." If the clusters tend to be somehow elongated or of a "chain" type nature, then this method is inappropriate.



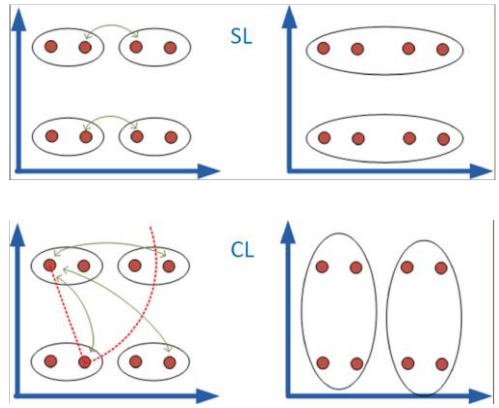
Hierarchical clustering

- Pair-group average. The distance between two clusters is calculated as the average distance between all pairs of objects in the two different clusters. This method is also very efficient when the objects form natural distinct "clumps," however, it performs equally well with elongated, "chain" type.



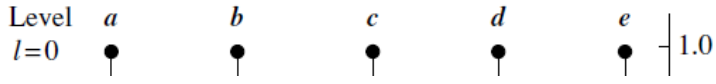
Hierarchical clustering

- Comparing single and complete link



Hierarchical clustering

- A tree structure called a dendogram is commonly used to represent the process of hierarchical clustering



l= A clustering of the data objects is obtained by
l= cutting the dendrogram at the desired level, then
each connected component forms a cluster
l=

Hierarchical clustering

- Measure for distance between clusters

Minimum distance: $dist_{min}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} \{|p - p'|\}$

Maximum distance: $dist_{max}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} \{|p - p'|\}$

Mean distance: $dist_{mean}(C_i, C_j) = |m_i - m_j|$

Average distance: $dist_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i, p' \in C_j} |p - p'|$

Hierarchical clustering

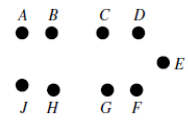
- Nearest-neighbor clustering algorithm: when an algorithm uses the minimum distance to measure the distance between clusters
- Single-linkage algorithm: the clustering process is terminated when the minimum distance between the nearest clusters exceeds a user-defined threshold
  - If we view the data points as nodes of a graph, with edges forming a path between the nodes in a cluster, then the merging of two clusters, C_i and C_j , corresponds to adding an edge between the nearest pair of nodes in C_i and C_j.
  - The resulting graph will generate a tree
  - Agglomerative hierarchical clustering algorithm that uses the minimum distance measure is also called minimal spanning tree algorithm

Hierarchical clustering

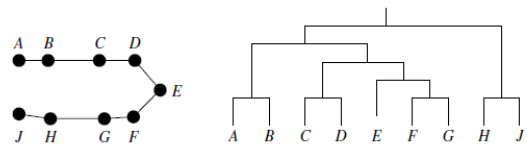
- Farthest-neighbor clustering algorithm: when an algorithm uses the maximum distance to measure the distance between clusters
- Complete-linkage algorithm: the clustering process is terminated when the maximum distance between nearest clusters exceeds a user-defined threshold
  - By viewing data points as nodes of a graph, with edges linking nodes, we can think of each cluster as a complete subgraph, that is, with edges connecting all the nodes in the clusters.
  - Tend to minimize the increase in diameter of the clusters at each iteration
  - High quality in case of clusters compact and of approximately equal size.

Hierarchical clustering

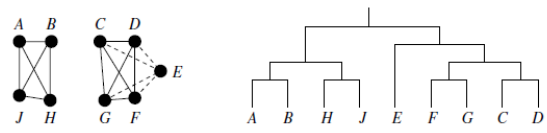
- Example



(a) Data set



(b) Clustering using single linkage



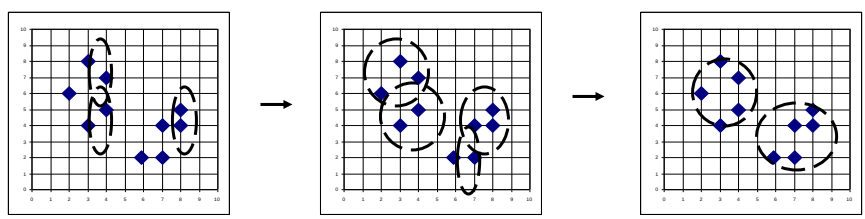
(c) Clustering using complete linkage

AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Use the single-link method and the dissimilarity matrix
- Initially, each object is placed into a cluster
  - Clusters are merged according to some criterion
    - For instance, if the distance between two objects belonging to two different clusters is the minimum distance between any two objects from different clusters (single-linkage approach)
  - The cluster merging process repeats until all of the objects are eventually merged to form one cluster

AGNES (Agglomerative Nesting)

- Single-linkage approach: each cluster is represented by all of the objects in the cluster, and the similarity between two clusters is measured by the similarity of the closest pair of data points belonging to different clusters.
- AGNES example



AGNES (Agglomerative Nesting)

	A	B	C	D	E
A	0	1	2	2	3
B	1	0	2	4	3
C	2	2	0	1	5
D	2	4	1	0	3
E	3	3	5	3	0

d = 1
A ∪ B
C ∪ D

Recompute the distance matrix

	AB	CD	E
AB	0	2	3
CD	2	0	3
E	3	3	0

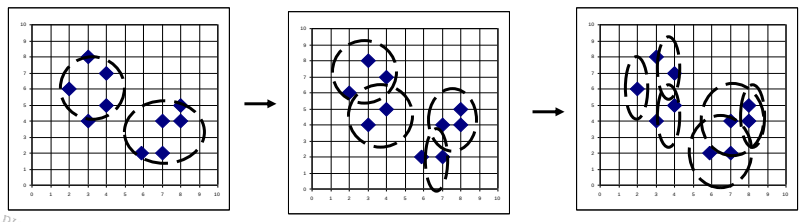
d = 2
(A ∪ B) ∪ (C ∪ D)

	ABCD	E
ABCD	0	3
E	3	0

d = 3
(ABCD) ∪ (E)

DIANA (Divisive Analysis)

- Introduced in Kaufmann and Rousseeuw (1990)
- Inverse order of AGNES
- Eventually each node forms a cluster on its own



DIANA (Divisive Analysis)

- The algorithm constructs a hierarchy of clusters, starting with one large cluster containing all n samples. Clusters are divided until each cluster contains only a single sample.
- At each stage, the cluster with the largest dissimilarity between any two of its samples is selected.
- To divide the selected cluster, the algorithm first looks for its most disparate sample (i.e., which has the largest average dissimilarity to the other observations of the selected cluster). This observation initiates the "splinter group". In subsequent steps, the algorithm reassigns observations that are closer to the "splinter group" than to the "old party". The result is a division of the selected cluster into two new clusters.

DIANA (Divisive Analysis)

The Algorithm:

1. Find the object, which has the **highest average dissimilarity** to all other objects. This object initiates a new cluster– a sort of a splinter group.
2. For each object i outside **the splinter group compute D_i** = [average d(i,j), $j \notin R_{splinter\ group}$] - [average d(i,j), $j \in R_{splinter\ group}$]
3. Find an object h for which **the difference D_h is the largest**. If D_h is positive, then h is, on the average close to the splinter group.
4. Repeat Steps 2 and 3 until all differences D_h are negative. The data set is then split into two clusters.

DIANA (Divisive Analysis)

The Algorithm:

5. Select the cluster with the largest diameter. The diameter of a cluster is the largest dissimilarity between any two of its objects. Then divide this cluster, following steps 1-4.
6. Repeat Step 5 until all clusters contain only a single object.

DIANA (Divisive Analysis)

An example:

X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
-0.308	-0.179	0.210	0.421	1.224	1.579	1.681	1.717

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
X_1	0	0.129	0.518	0.729	1.532	1.887	1.989	2.025
X_2	0.129	0	0.389	0.6	1.403	1.758	1.86	1.896
X_3	0.518	0.389	0	0.211	1.014	1.369	1.471	1.507
X_4	0.729	0.6	0.211	0	0.803	1.158	1.26	1.296
X_5	1.532	1.403	1.014	0.803	0	0.355	0.457	0.493
X_6	1.887	1.758	1.369	1.158	0.355	0	0.102	0.138
X_7	1.989	1.86	1.471	1.26	0.457	0.102	0	0.036
X_8	2.025	1.896	1.507	1.296	0.493	0.138	0.036	0

DIANA (Divisive Analysis)

An example:

We need to find the most dissimilar element, thus we calculate the mean for each row:

Ex: $d(1,A_0) = (0.129+0.518+0.729+1.532+1.887+1.989+2.025) / 7 = 1.258$

A_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
$d(i,A_0)$	1.258	1.147	0.925	0.865	0.865	0.966	1.025	1.056

$A_1 = \{2,3,4,5,6,7,8\}$	X_6 X_7 X_8 X_5 X_4 X_3 X_2
$B_1 = \{1\}$	X_1

DIANA (Divisive Analysis)

An example:

We need to find the most dissimilar element for the cluster A_1, thus we calculate the mean for each row (the X_1-column and X_1-row are not considered):

Ex: $d(1,A_1) = (0.389+0.6+1.403+1.758+1.86+1.896) / 6 = 1.318$

A_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
$d(i,A_1)$	1.318	0.993	0.889	0.754	0.813	0.864	0.894

In this case: $d(X_2,A_1) = 1.318$ and $d(X_2,B_1) = 0.129$

$A_2 = \{3,4,5,6,7,8\}$	X_6 X_7 X_8 X_5 X_4 X_3
$B_2 = \{1,2\}$	X_1 X_2

DIANA (Divisive Analysis)

An example:

And so on:

A_2	X_3	X_4	X_5	X_6	X_7	X_8
$d(i,A_2)$	1.114	0.946	0.624	0.624	0.665	0.694

B_2	X_1	X_2
$d(i,B_2)$	0.129	0.129

In this case: $d(X_3,A_2) = 1.114$ and $d(X_3,B_2) = 0.453$, thus the element X_3 is included in B.

$A_3 = \{4,5,6,7,8\}$	X_6 X_7 X_8 X_5 X_4	X_1 X_2 X_3
$B_3 = \{1,2,3\}$		

88

DIANA (Divisive Analysis)

An example:

A_3	X_4	X_5	X_6	X_7	X_8
$d(i,A_3)$	1.129	0.527	0.438	0.464	0.491

B_3	X_1	X_2	X_3
$d(i,B_3)$	0.323	0.259	0.453

In this case: $d(X_4,A_3) = 1.129$ and $d(X_4,B_3) = 0.513$, thus the element X_4 is included in B.

$A_4 = \{5,6,7,8\}$	X_6 X_7 X_8 X_5	X_1 X_2 X_3 X_4
$B_4 = \{1,2,3,4\}$		

89

89

DIANA (Divisive Analysis)

In order to choose which cluster to divide,

diam. A_4 = max d(i,j) = d(5,8) = 0.493 i,j ∈ A_4

diam. B_4 = max d(i,j) = d(1,4) = 0.729 i,j ∈ B_4

Diam B_4>diam A_4, thus we are going to divide the cluster B_4

B_4	X_1	X_2	X_3	X_4
d(i,B_4)	0.459	0.373	0.373	0.513

$A_5 = \{5,6,7,8\}$	X_6 X_7 X_8 X_5 X_4 X_1 X_2 X_3
$B_5 = \{1,2,3\}$	
$C_5 = \{4\}$	

DIANA (Divisive Analysis)

B_5	X_1	X_2	X_3
d(i,B_5)	0.323	0.259	0.453

$d(X_3,B_5) = 0.453$, $d(X_3,C_5) = 0.211$

$A_6 = \{5,6,7,8\}$	X_6 X_7 X_8 X_5 X_3 X_4 X_1 X_2
$B_6 = \{1,2\}$	
$C_6 = \{3,4\}$	

A_6	X_5	X_6	X_7	X_8
d(i,A_6)	0.435	0.198	0.198	0.222

DIANA (Divisive Analysis)

B_6	X_1	X_2
$d(i,B_6)$	0.129	0.129

C_6	X_3	X_4
$d(i,C_6)$	0.211	0.211

The maximum distance is $d(X_5,A_6) = 0.435$ and $d(X_5,B_6) = 1.467$ and $d(X_5,C_6) = 1.817$; thus the element X_5 can not be included in B or C, and the process stops.

$$\text{diam. } A_6 = \max d(i,j) = d(5,8) = 0.493 \qquad i,j \in A_6$$
$$\text{diam. } B_6 = \max d(i,j) = d(1,2) = 0.129 \qquad i,j \in B_6$$
$$\text{diam. } C_6 = \max d(i,j) = d(3,4) = 0.211 \qquad i,j \in C_6$$

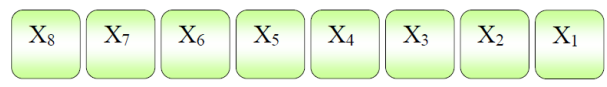
diam. A_6 > diam. C_6 > diam. B_6, thus we are going to divide the cluster A_6 creating the *cluster* $\{5\}$

92

DIANA (Divisive Analysis)

The final result is:

$A_{11} = \{7\} = 1.681$
$B_{11} = \{1\} = -0.308$
$C_{11} = \{3\} = 0.210$
$D_{11} = \{4\} = 0.421$
$E_{11} = \{2\} = -0.179$
$F_{11} = \{5\} = 1.224$
$G_{111} = \{6\} = 1.579$
$H_{11} = \{8\} = 1.717$



93

Hierarchical Clustering

- Major weaknesses of hierarchical clustering methods
  - Can never undo what was done previously
  - Do not scale well: time complexity of at least O(n^2), where n is the number of total objects
- Major strengths
  - It's nice that you get a hierarchy instead of an amorphous collection of groups
  - Don't need to specify k
    - If you want k groups, just cut the (k-1) longest links
  - In general give better quality clusters than k-means' like methods

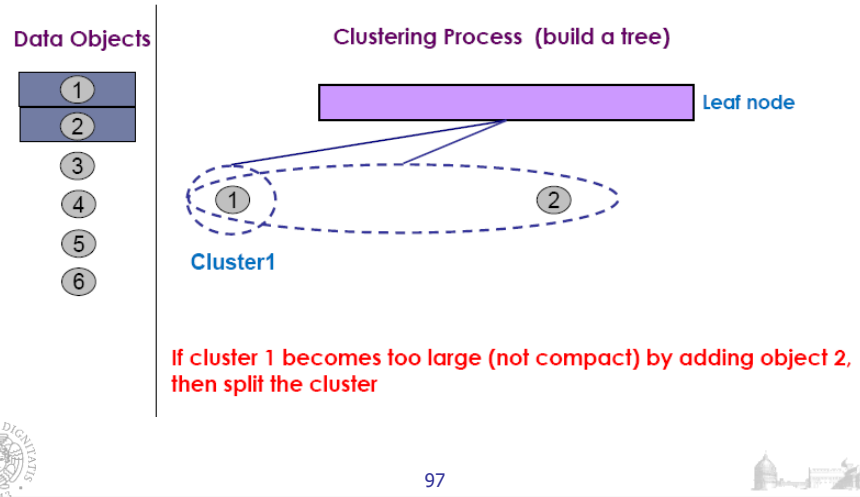
Extensions to Hierarchical Clustering

- Integration of hierarchical and distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

BIRCH

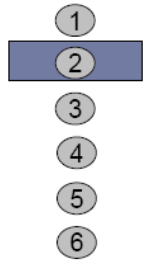
- BIRCH: Balanced Iterative Reducing and Clustering Using Hierarchies
- **Agglomerative Clustering** designed for clustering a large amount of numerical data
- What does BIRCH algorithm try to solve?
  - Most of the existing algorithms DO NOT consider the case that datasets can be too large to fit in main memory
  - They DO NOT concentrate on minimizing the number of scans of the dataset
  - I/O costs are very high
- The complexity of BIRCH is O(n) where n is the number of objects to be clustered.

BIRCH

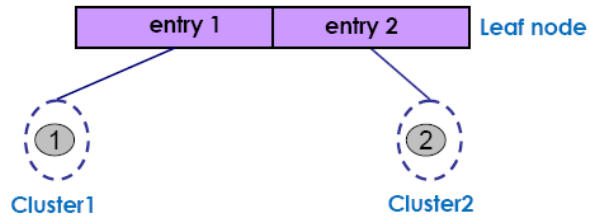


BIRCH

Data Objects



Clustering Process (build a tree)

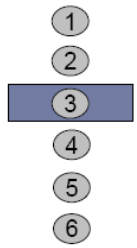


Leaf node with two entries

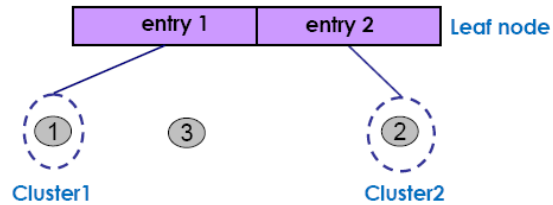
98

BIRCH

Data Objects



Clustering Process (build a tree)



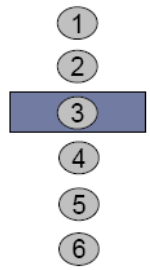
entry1 is the closest to object 3

If cluster 1 becomes too large by adding object 3, then split the cluster

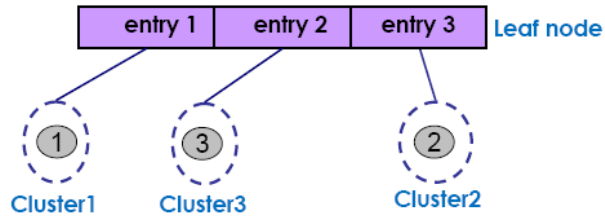
99

BIRCH

Data Objects



Clustering Process (build a tree)

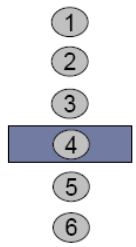


Leaf node with three entries

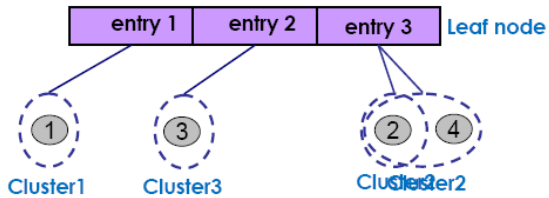
100

BIRCH

Data Objects



Clustering Process (build a tree)



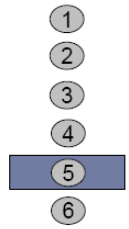
entry3 is the closest to object 4

Cluster 2 remains compact when adding object 4
then add object 4 to cluster 2

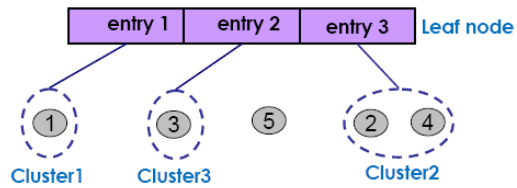
101

BIRCH

Data Objects



Clustering Process (build a tree)



entry2 is the closest to object 5

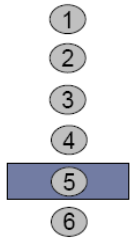
Cluster 3 becomes too large by adding object 5
then split cluster 3?

BUT there is a limit to the number of entries a node can have
Thus, split the node

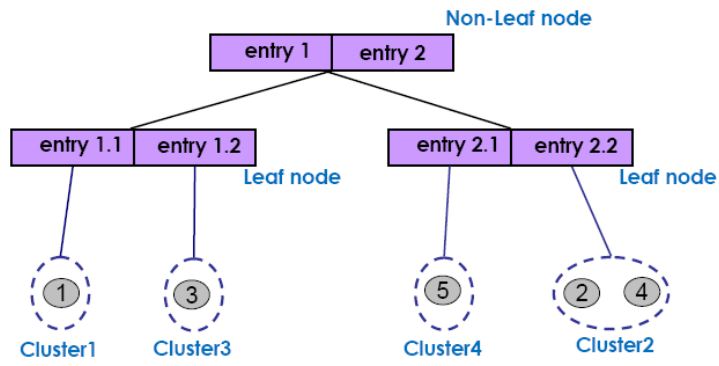
102

BIRCH

Data Objects

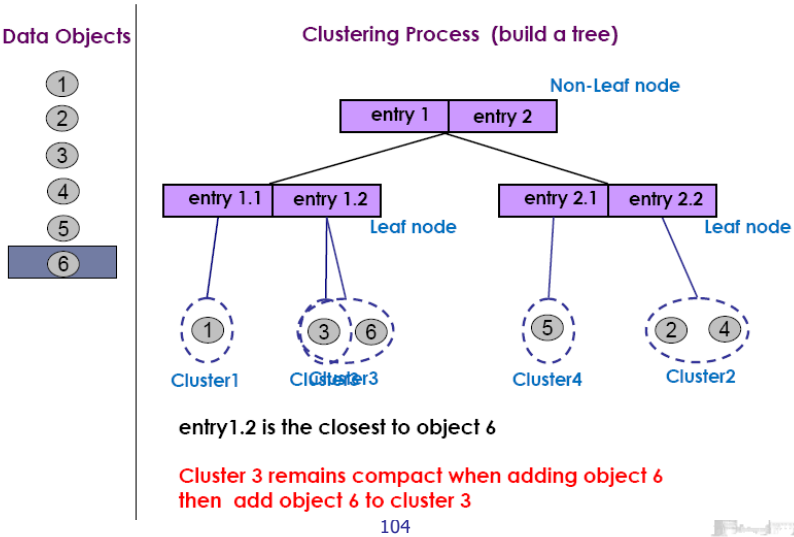


Clustering Process (build a tree)



103

BIRCH



entry1.2 is the closest to object 6

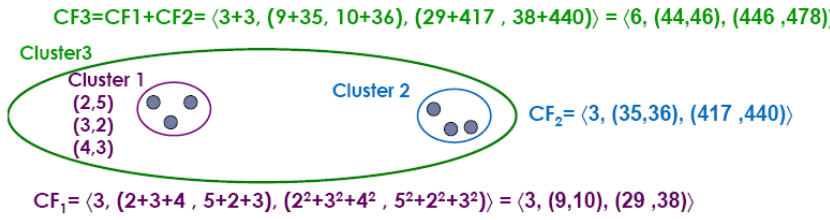
Cluster 3 remains compact when adding object 6 then add object 6 to cluster 3

BIRCH: Key components

- Clustering Feature (CF= (N,LS,SS))
  - Summary of the statistics for a given cluster: the 0-th, 1st and 2nd moments of the cluster from the statistical point of view
  - Used to compute centroids, and measure the compactness and distance of clusters

Clustering Features

- N: number of data points
- LS: linear sum of N points $LS = \sum_{i=1}^{n} x_i$
- SS: square sum of N points $SS = \sum_{i=1}^{n} x_i^2$



BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies)

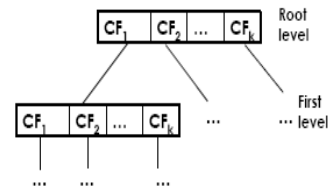
- CF entry is a summary of statistics of the cluster and has sufficient information to calculate the centroid, radius, diameter and many other measures
- Centroid: the "middle" of a cluster $x_0 = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{LS}{n}$
- Radius: square root of average distance from any point of the cluster to its centroid $R = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x_0)^2}{n}} = \sqrt{\frac{nSS - 2LS^2 + nLS^2}{n^2}}$
- Diameter: square root of average mean squared distance between all pairs of points in the cluster $D = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n} (x_i - x_j)^2}{n(n-1)}} = \sqrt{\frac{2nSS - 2LS^2}{n(n-1)}}$

BIRCH: Key components

- CF-Tree
  - Height-balance tree
  - Two parameters
    - Number of entries in each node
    - The diameter of all entries in a leaf node
  - Leaf nodes are connected via prev and next pointers

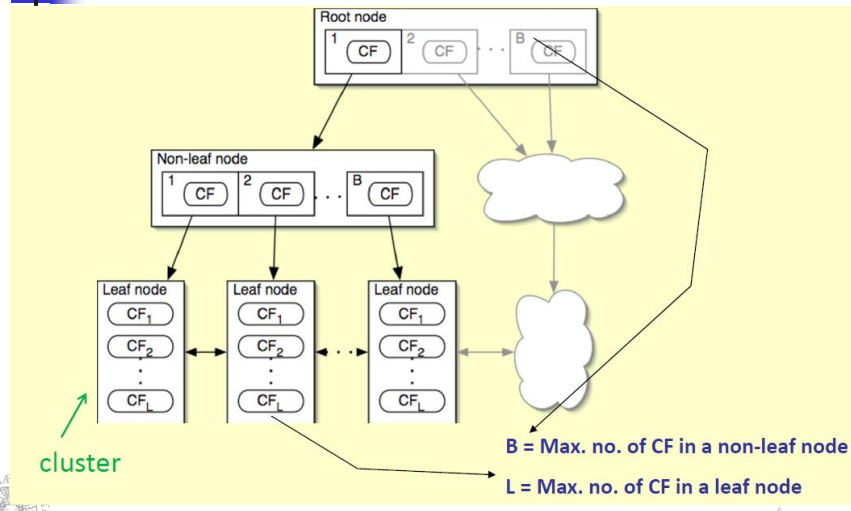
BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies)

- A CF-tree is a height-balanced tree that stores the clustering features CFs for a hierarchical clustering,



- Parameters:
  - B = Branching factor specifies the maximum number of children per nonleaf node.
  - T = Threshold parameter specifies the maximum diameter of subclusters stored at the leaf nodes of the tree.
- L = Max. number of entries in a leaf
- CF entry in parent = sum of CF entries of a child of that entry

The CF Tree Structure



BIRCH – Notes

- A Leaf node represents a cluster.
- A sub-cluster in a leaf node must have a diameter no greater than a given threshold T.
- A point is inserted into the leaf node (cluster) to which is closer.
- When one item is inserted into a cluster at the leaf node, the restriction T (for the corresponding subcluster) must be satisfied. The corresponding CF must be updated.
- If there is no space on the node the node is split.

BIRCH algorithm

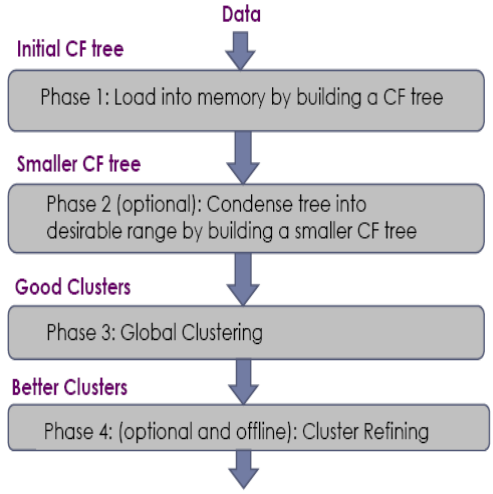
- Incrementally construct a CF tree, a hierarchical data structure for multiphase clustering
- Phase 1: scan DB to build an initial in-memory CF tree
  - If threshold condition is violated
    - If there is room to insert – Insert point as a single cluster
    - If not
      - Leaf node split: take two farthest CFs and create two leaf nodes, put the remaining CFs (including the new one) into the closest node
      - Update CF for non-leafs. Insert new non-leaf entry into parent node
      - We may have to split the parent as well. Split the root increases tree height by one.

BIRCH algorithm

  - If not
    - Insert point into the closest cluster

- Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree

BIRCH Algorithm



BIRCH Algorithm: Phase 1

- Phase 1
  - Choose an initial value for threshold, start inserting the data points one by one into the tree as per the insertion algorithm
  - If, in the middle of the above step, the size of the CF tree exceeds the size of the available memory, increase the value of threshold
  - Convert the partially built tree into a new tree
    - The rebuild process is performed by building a new tree from the leaf nodes of the old tree.
    - No need of rereading all the objects
  - Repeat the above steps until the entire dataset is scanned and a full tree is built
  - Outlier Handling

BIRCH Algorithm: Phase 2, 3 and 4

- Phase 2
  - A bridge between phases 1 and 3
  - Builds a smaller CF tree by increasing the threshold
- Phase 3
  - Apply global clustering algorithm to the sub-clusters given by leaf entries of the CF tree
  - Improves clustering quality
- Phase 4
  - Scan the entire dataset to label the data points
  - Outlier handling

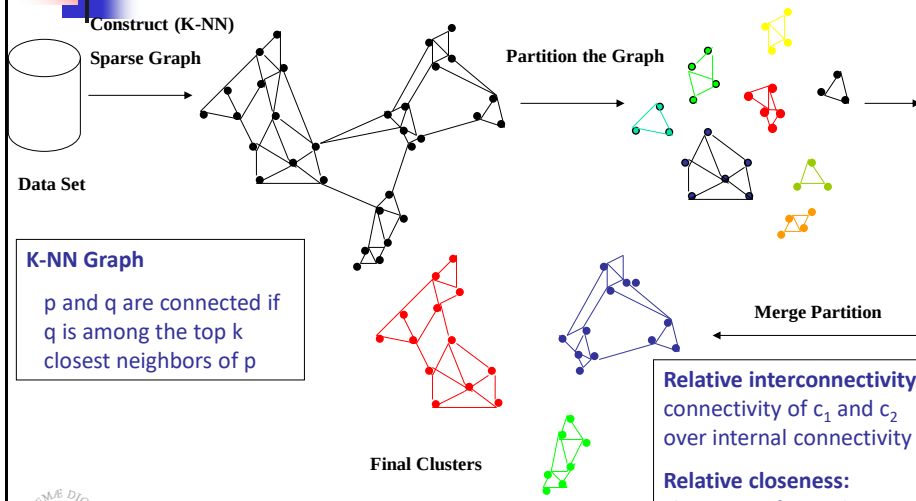
BIRCH Algorithm

- Strengths:
  - finds a good clustering with a single scan and improves the quality with a few additional scans
  - Complexity is *O(n)*
- Weakness:
  - **Handles only numeric data**, and sensitive to the order of the data record
  - **Sensitive to insertion order of data points**
  - Since we fix the size of leaf nodes, so clusters may not be so natural
  - **Clusters tend to be spherical** given the radius and diameter measures

CHAMELEON: Hierarchical Clustering Using Dynamic Modeling (1999)

- CHAMELEON: G. Karypis, E. H. Han, and V. Kumar, 1999
- Measures the similarity based on a dynamic model
  - Two clusters are merged only if the interconnectivity and closeness (proximity) between two clusters are high relative to the internal interconnectivity of the clusters and closeness of items within the clusters
- Graph-based, and a two-phase algorithm
  - Use a graph-partitioning algorithm: cluster objects into a large number of relatively small sub-clusters
  - Use an agglomerative hierarchical clustering algorithm: find the genuine clusters by repeatedly combining these sub-clusters

Overall Framework of CHAMELEON



K-NN Graph

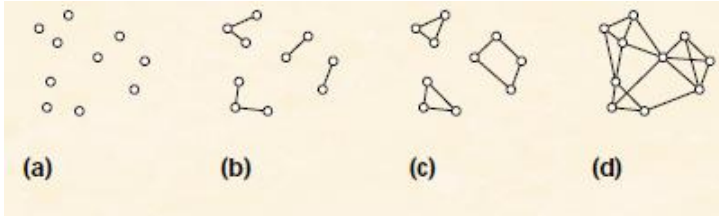
p and q are connected if q is among the top k closest neighbors of p

Relative interconnectivity: connectivity of c_1 and c_2 over internal connectivity

Relative closeness: closeness of c_1 and c_2 over internal closeness

CHAMELEON: the effect of k

- The effect of k
  - b -> k=1
  - c -> k=2
  - d -> k=3



CHAMELEON: discussion

- The k-nearest neighbor captures the concept of neighborhood dynamically
  - The neighborhood radius of an object is determined by the density of the region in which the object resides
  - The density of the region is recorded as the weight of the edges: the edges of a dense region tend to weigh more than those of a sparse region. In particular, the weight of each edge represents the closeness between two samples, that is, an edge will weigh more whether the two data samples are closer to each other.
  - More natural clusters than DBSCAN

CHAMELEON: discussion

- Graph-partitioning algorithm:
  - A cluster C is partitioned into subclusters C_i and C_j so as to minimize the sum of the weights (edge cut) of the edges that would be cut should C be bisected into C_i and C_J.
  - Each one of these sub-clusters contains at least 25% of the nodes in C.
  - Edge cut $EC_{\{Ci,Cj\}}$ assesses the absolute interconnectivity between subclusters C_i and C_J, that is, the sum of the weight of the edges that connect vertices in C_i to vertices in C_j.

CHAMELEON: discussion

CHAMELEON obtains the initial set of sub-clusters as follows.

- Starts with **all the points belonging to the same cluster**.
- Then repeatedly **selects the largest sub-cluster** among the current set of sub-clusters and uses the graph-partitioning algorithm to bisect it.
- This process terminates when the **larger sub-cluster contains fewer than a specified number of vertices** (1% to 5% of the overall number of data)

CHAMELEON: discussion

- The agglomerative hierarchical clustering algorithm merges subclusters based on their similarity
- The similarity is determined according to their relative interconnectivity, $RI(C_i, C_j)$ and their relative closeness, $RC(C_i, C_j)$.

$$RI(C_i, C_j) = \frac{|EC_{\{C_i,C_j\}}|}{\frac{1}{2}(|EC_{C_i}| + |EC_{C_j}|)}$$

where EC_{Ci} is the internal inter-connectivity of a cluster C_i (i.e., the minimum sum of the weights of edges which are cut by partitioning C_i into two roughly equal parts).

CHAMELEON: discussion

- The relative closeness $RC(C_i, C_j)$ is defined as:

$$RC(C_i, C_j) = \frac{\overline{S}_{EC_{\{C_i,C_j\}}}}{\frac{|C_i|}{|C_i|+|C_j|}\overline{S}_{EC_{C_i}} + \frac{|C_j|}{|C_i|+|C_j|}\overline{S}_{EC_{C_j}}}$$

where $\overline{S}_{EC_{\{C_i,C_j\}}}$ is the average weight of the edges that connect vertices in C_i to vertices in C_j and $\overline{S}_{EC_{C_i}}$ is the average weight of the edges that belong to the min-cut bisector of cluster C_i.

- The agglomerative hierarchical clustering merges only those pairs of clusters whose relative inter-connectivity and relative closeness are both above some user specified threshold T_{RI} and T_{RC}, respectively.

CHAMELEON: discussion

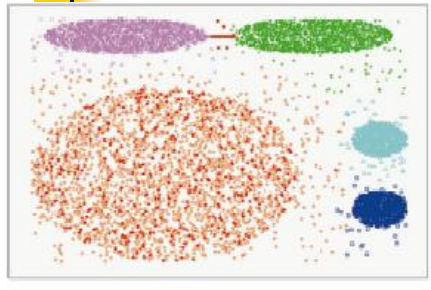
- CHAMELEON visits each cluster C_i , and checks to see if any one of its adjacent clusters C_j satisfy the following two conditions:
  - If more than one of the adjacent clusters satisfy the conditions, then CHAMELEON selects to merge C_i with the cluster that it is most connected to; i.e., it selects the cluster C_j such that the absolute inter-connectivity between these two clusters is the highest.

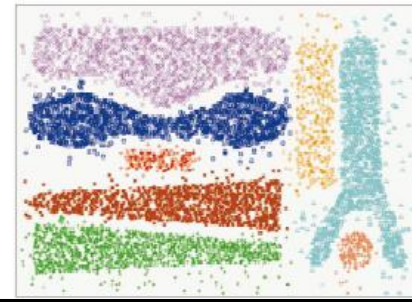
$$RI(C_i, C_j) \geq T_{RI} \text{ and } RC(C_i, C_j) \geq T_{RC}$$

- Complexity $O(n^2)$ (in the worst case) where n is the number of objects
- Greater power at discovering arbitrarily shaped clusters of high quality than several well-known algorithms as BIRCH and DBSCAN

126

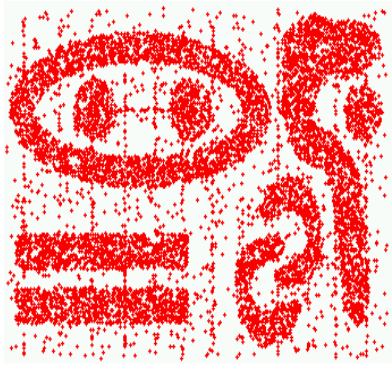
CHAMELEON: discussion



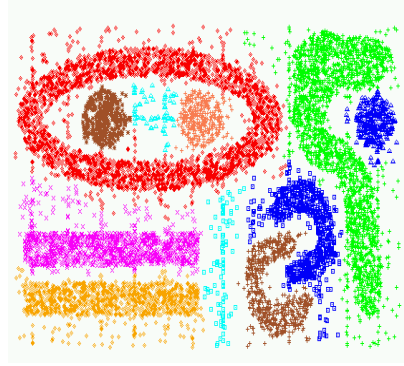


127

CHAMELEON: discussion



Original dataset



Chameleon

Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

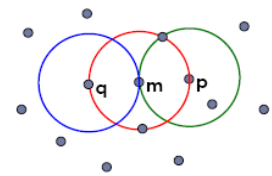
Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as density-connected points
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition
- Several interesting studies:
  - DBSCAN: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - DENCLUE: Hinneburg & D. Keim (KDD'98)
  - CLIQUE: Agrawal, et al. (SIGMOD'98) (more grid-based)

Density-Based Clustering: Basic Concepts

- Two parameters:
  - *Eps*: Maximum radius of the neighbourhood
  - *MinPts*: Minimum number of points in an Eps-neighbourhood of that point
- If the Eps-neighborhood (neighborhood within radius Eps - N_{Eps}) of an object contains at least a minimum number, MinPts, of objects then the object is called core object.
- Example: eps = 1 cm, MinPts=3

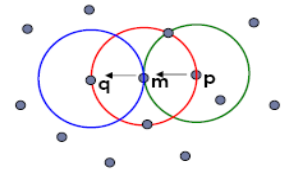
  m and p are core objects



Density-Based Clustering: Basic Concepts

- Directly density-reachable: An object *p* is directly density-reachable from an object *q* w.r.t. *Eps*, *MinPts* if
  - *p* belongs to $N_{Eps}(q)$
  - *q* is a core object, that is,

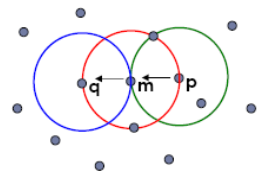
    $|N_{Eps}(q)| \geq MinPts$



- Example
  - q is directly density-reachable from m
  - m is directly density-reachable from p and vice versa

Density-Reachable

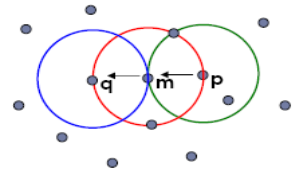
- Density-reachable:
  - An object *p* is density-reachable from an object *q* w.r.t. *Eps*, *MinPts* if there is a chain of objects $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i



- Example
  - q is density-reachable from p because q is directly density-reachable from m and m is directly density-reachable from p
  - p is not density-reachable from q because q is not a core object

Density-Connected

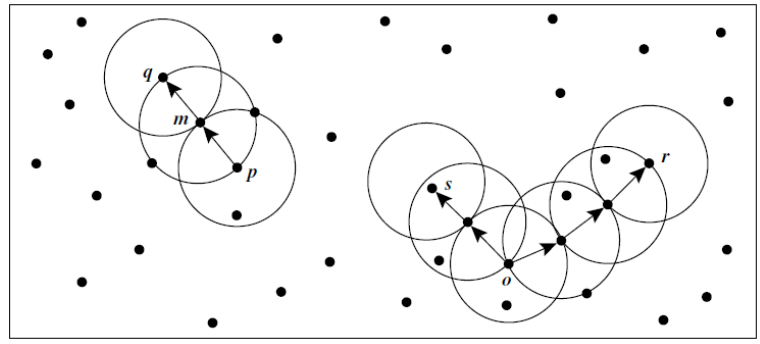
- Density-connected
  - An object *p* is density-connected to an object *q* w.r.t. *Eps*, *MinPts* if there is an object *o* such that both *p* and *q* are density-reachable from *o* w.r.t. *Eps* and *MinPts*



- Example:
  - p, q and m are all density-connected

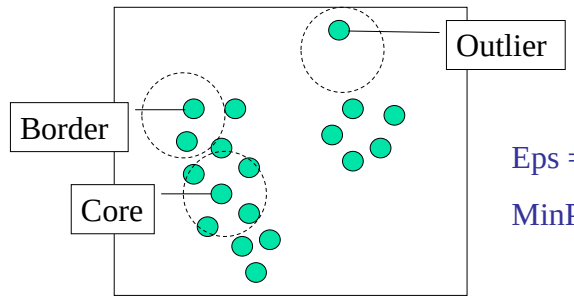
Density-Reachable and Density-Connected

- Object *q* is (indirectly) density-reachable from *p* because *q* is directly density reachable from *m* and *m* is directly density-reachable from *p*. However, *p* is not density reachable from *q* because *q* is not a core object. Similarly, *r* and *s* are density-reachable from *o* and *o* is density-reachable from *r*. Thus, *o*, *r*, and *s* are all density-connected.



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: Density-Based Spatial Clustering of Applications with Noise

- Searches for clusters by checking the Eps-neighborhood of each object in the database
- If the Eps-neighborhood of an object p contains more than MinPts, a new cluster with a core object is created
- DBSCAN iteratively collects directly density reachable objects from these core objects: this may involve the merge of a few density-reachable clusters
- The process terminated when no new point can be added to any cluster.

DBSCAN: Density-Based Spatial Clustering of Applications with Noise

- DBSCAN adopts the closure of density-connectedness to find connected dense regions as clusters. Each closed set is a density-based cluster.
- A subset C in D is a cluster if
  - For any two objects o_1, o_2 in C, o_1 and o_2 are density-connected, and
  - there does not exist an object o in C and another object o' in (D-C) such that o and o' are density-connected

DBSCAN: Density-Based Spatial Clustering of Applications with Noise

- Complexity: $O(n^2)$

Algorithm: DBSCAN: a density-based clustering algorithm.

Input:

- D: a data set containing n objects,
- ϵ: the radius parameter, and
- *MinPts*: the neighborhood density threshold.

Output: A set of density-based clusters.

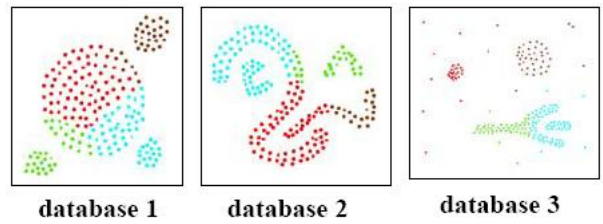
DBSCAN: Density-Based Spatial Clustering of Applications with Noise

Method:

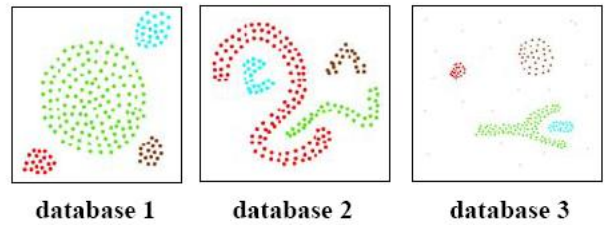
```
(1)  mark all objects as unvisited;
(2)  do
(3)      randomly select an unvisited object p;
(4)      mark p as visited;
(5)      if the ε-neighborhood of p has at least MinPts objects
(6)          create a new cluster C, and add p to C;
(7)          let N be the set of objects in the ε-neighborhood of p;
(8)          for each point p' in N
(9)              if p' is unvisited
(10)                 mark p' as visited;
(11)                 if the ε-neighborhood of p' has at least MinPts points,
                     add those points to N;
(12)             if p' is not yet a member of any cluster, add p' to C;
(13)         end for
(14)         output C;
(15)     else mark p as noise;
(16) until no object is unvisited;
```


DBSCAN: Sensitive to Parameters

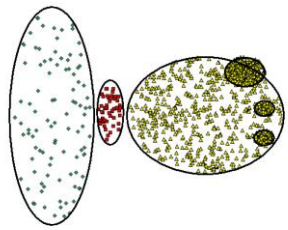
CLARANS



DBSCAN

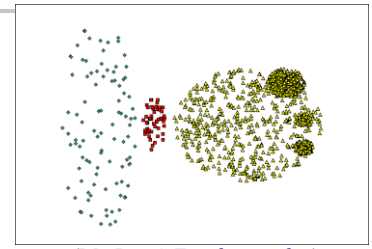


DBSCAN: Sensitive to Parameters

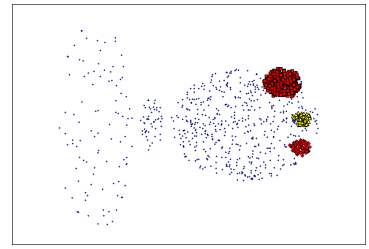


Original Points

- Varying densities
- High-dimensional data



(MinPts=4, Eps=large value)



(MinPts=4, Eps=small value; min density increases)

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

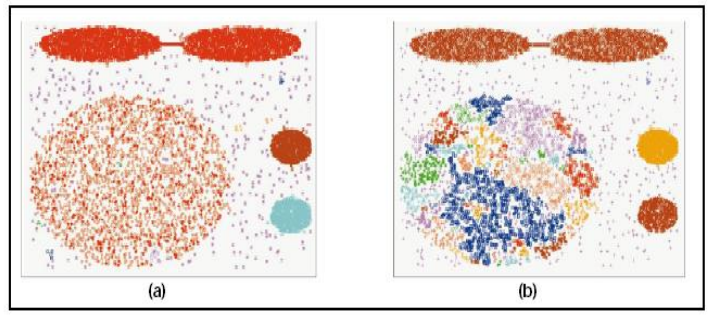
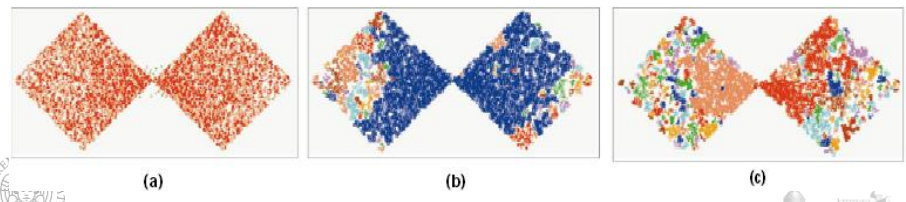


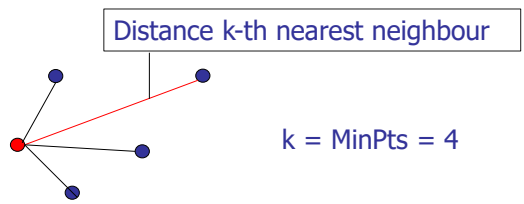
Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



How to determine Eps and MinPts

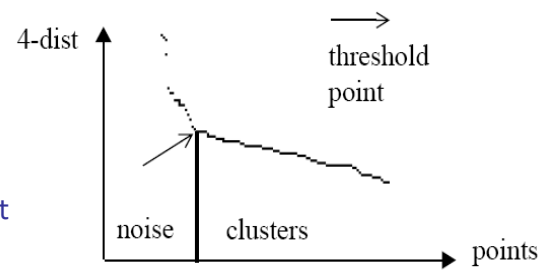
Heuristic approach

- For a given k we define a function k-dist, mapping each point to the distance from its k-th nearest neighbor.



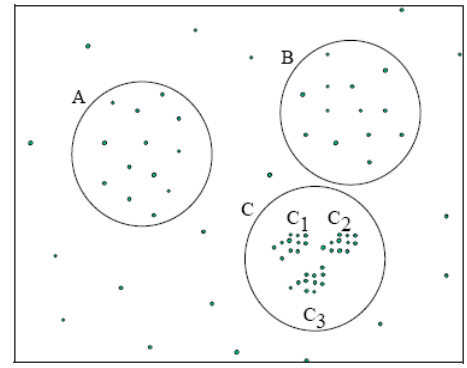
How to determine Eps and MinPts

- We sort the points in descending order of their k-dist values: the graph of this function gives some hints concerning the density distribution.
- If we choose an arbitrary point p, set the parameter Eps to k-dist(p) and set the parameter MinPts to k, all points with an equal or smaller k-dist value will be core points.
- The threshold point is the first point in the first "valley" of the sorted k-dist graph



OPTICS: A Cluster-Ordering Method (1999)

- Problem: the intrinsic cluster structure cannot be characterized by global density parameters.
- First alternative
  - to use a hierarchical clustering algorithm, for instance the single-link method.
  - Drawbacks:
    - single-link effect, i.e. clusters which are connected by a line of few points having a small inter-object distance are not separated.
    - the results, i.e. the dendrograms, are hard to understand for more than a few hundred objects.



OPTICS: A Cluster-Ordering Method (1999)

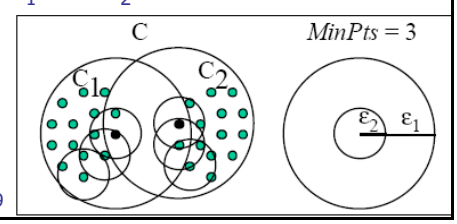
- Second alternative
  - to use a density-based partitioning algorithm with different parameter settings.
  - Drawbacks
    - there are an infinite number of possible parameter values.
    - Even if we use a very large number of different values - which requires a lot of secondary memory to store the different cluster memberships for each point - it is not obvious how to analyze the results and we may still miss the interesting clustering levels.
- Solution
  - to run an algorithm which produces a special order of the database with respect to its density-based clustering structure containing the information about every clustering level of the data set (up to a "generating distance" eps), and is very easy to analyze.

OPTICS: A Cluster-Ordering Method (1999)

- OPTICS: Ordering Points To Identify the Clustering Structure
  Ankerst, Breunig, Kriegel, and Sander (SIGMOD'99)
  - Produces a special order of the database with respect to its density-based clustering structure
  - This cluster-ordering contains info equivalent to the density-based clustering corresponding to a broad range of parameter settings
  - Good for both automatic and interactive cluster analysis, including finding intrinsic clustering structure
  - Can be represented graphically or using visualization techniques

OPTICS: A Cluster-Ordering Method (1999)

- Observation: density-based clusters are monotonic with respect to the neighborhood threshold
- DBSCAN – for a constant MinPts value, density-based clusters with respect to a higher density (i.e., a lower value of eps) are completely contained in density-connected sets obtained with respect to a lower density.
  - C_1 and C_2 are density-based clusters with respect to $eps_2 < eps_1$ and C is a density-based cluster with respect to eps_1 completely containing the sets C_1 and C_2

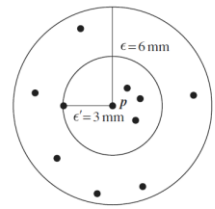


OPTICS: A Cluster-Ordering Method (1999)

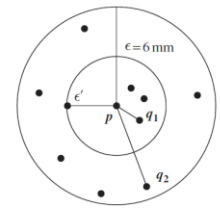
- IDEA
  - to process a set of distance parameter values at the same time. In practice, to use DBSCAN for an infinite number of distance parameters eps_i which are smaller than a "generating distance eps".
  - Unlike DBSCAN, OPTICS does not assign cluster memberships. Instead, it stores the order in 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).
    - This information consists of only two values for each object: the core-distance and a reachability-distance.

OPTICS: A Cluster-Ordering Method (1999)

- Core-distance of an object p:
  - the smallest eps value that makes p a core object. If p is not a core object, the core-distance is undefined
- Reachability-distance of an object q from p is the minimum radius value that makes q directly density-reachable from p
  - p has to be a core object and q must be in the neighborhood of p. Therefore, the reachability-distance from p to q is max(core-distance(p), dist(p, q)). If p is not a core object, the reachability-distance is undefined



Core-distance of p



Reachability-distance $(p, q_1) = \epsilon' = 3$ mm
Reachability-distance $(p, q_2) = dist\,(p, q_2)$

OPTICS: A Cluster-Ordering Method (1999)

- To construct the different partitions simultaneously, the objects have to be processed in a specific order:
- OPTICS begins with an arbitrary object from the input database as the current object, p. It retrieves the eps-neighborhood of p, determines the core-distance, and sets the reachability-distance to undefined.
  - **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).
  - **If p is a core object**, then for each object q in the eps-neighborhood of p, OPTICS updates its reachability-distance from p and inserts q into OrderSeeds if q has not yet been processed.

152

OPTICS: A Cluster-Ordering Method (1999)

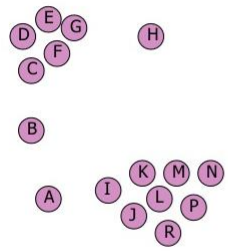
  - **The objects contained in OrderSeeds are sorted by their reachability-distance** to the closest core object from which they have been directly density reachable. In each step of the WHILE-loop, an object currentObject having the smallest reachability-distance in the seed-list is selected by the method OrderSeeds:next().
  - The eps-neighborhood of this object and its core-distance are determined. Then, the object is simply written to the file OrderedFile with its core distance and its current reachability-distance.
  - The iteration continues until the input is fully consumed and OrderSeeds is empty.

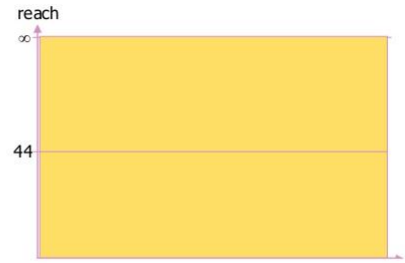
- At the end of the execution of OPTICS, we have the list of objects in the order in which the objects themselves are processed.
- Complexity: in the worst case, $O(n^2)$, where n is the number of objects to be clustered.

153

OPTICS: A Cluster-Ordering Method (1999)

- Example (Anu Singha, Asiya Naz, Rajesh Piryani, South Asian Univ.)
  - Example Database (2-dimensional, 16 points)
  - $\varepsilon = 44$, $MinPts = 3$



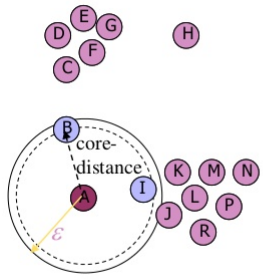


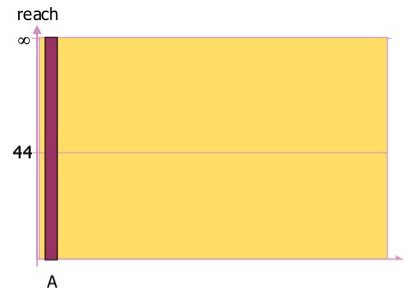
seedlist:

154

OPTICS: A Cluster-Ordering Method (1999)

- Example (Anu Singha, Asiya Naz, Rajesh Piryani, South Asian Univ.)



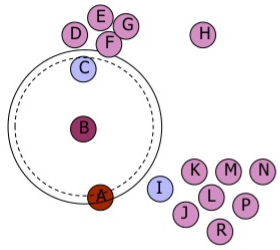


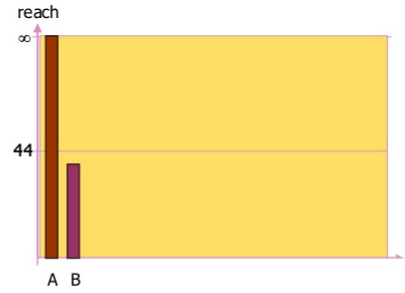
seedlist: (B,40) (I, 40)

155

OPTICS: A Cluster-Ordering Method (1999)

- Example (Anu Singha, Asiya Naz, Rajesh Piryani, South Asian Univ.)



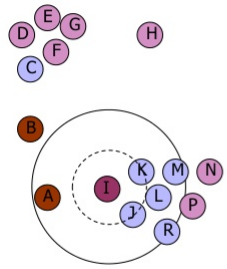


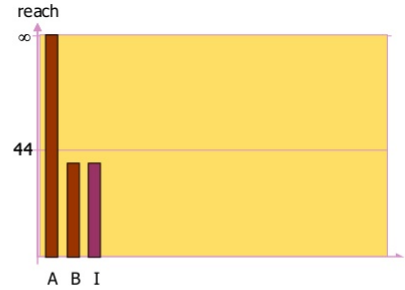
seedlist: (I, 40) (C, 40)

156

OPTICS: A Cluster-Ordering Method (1999)

- Example (Anu Singha, Asiya Naz, Rajesh Piryani, South Asian Univ.)



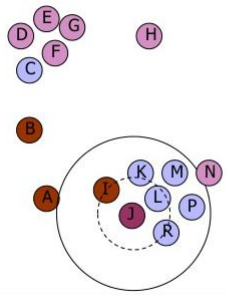


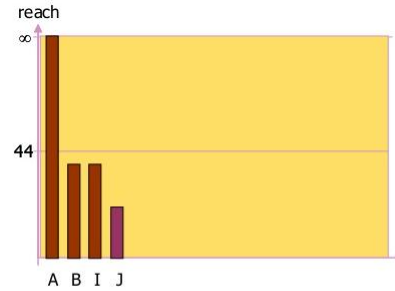
seedlist: (J, 20) (K, 20) (L, 31) (C, 40) (M, 40) (R, 43)

157

OPTICS: A Cluster-Ordering Method (1999)

- Example



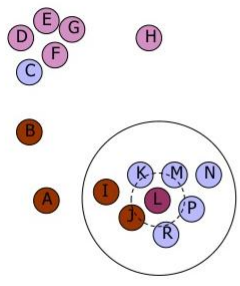


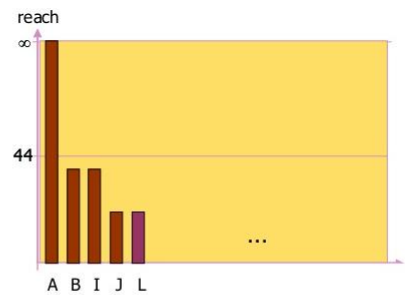
seedlist: (L, 19) (K, 20) (R, 21) (M, 30) (P, 31) (C, 40)

158

OPTICS: A Cluster-Ordering Method (1999)

- Example (Anu Singha, Asiya Naz, Rajesh Piryani, South Asian Univ.)



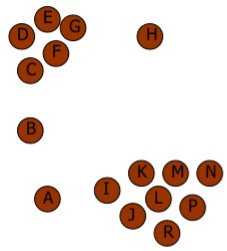


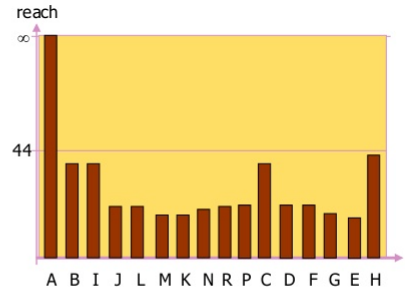
seedlist: (M, 18) (K, 18) (R, 20) (P, 21) (N, 35) (C, 40)

159

OPTICS: A Cluster-Ordering Method (1999)

- Example



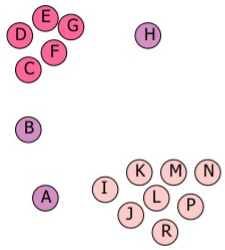


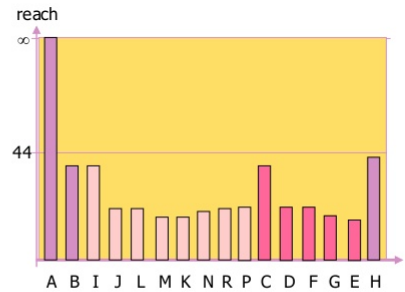
seedlist: -

160

OPTICS: A Cluster-Ordering Method (1999)

- Example (Anu Singha, Asiya Naz, Rajesh Piryani, South Asian Univ.)



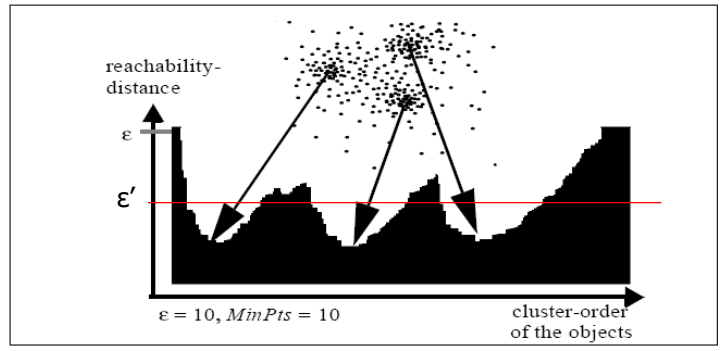


seedlist: -

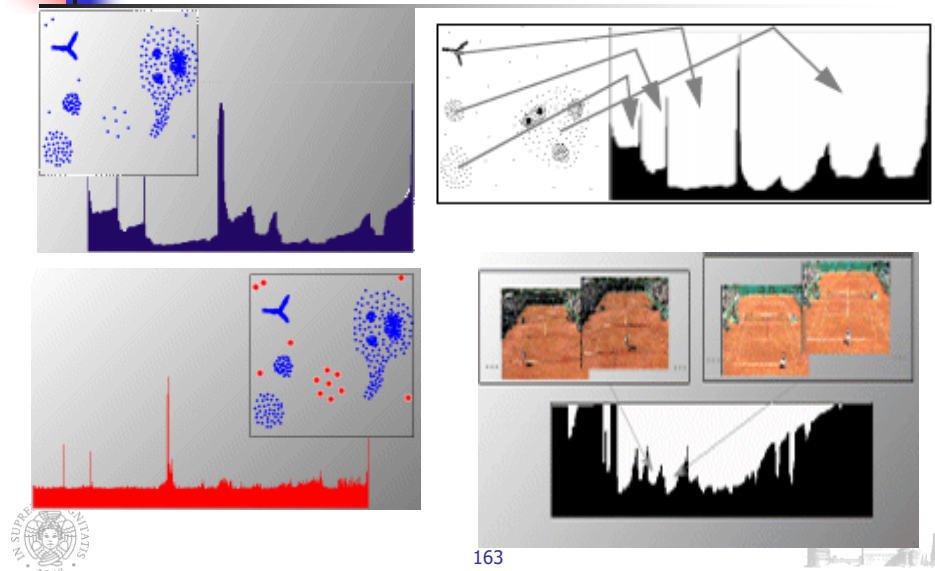
161

OPTICS: A Cluster-Ordering Method (1999)

- How are these values used?



Density-Based Clustering: OPTICS & Its Applications



DENCLUE: Using Statistical Density Functions

- DENsity-based CLUstEring by Hinneburg & Keim (KDD'98)
  - Problem of DBSCAN and OPTICS: density is calculated by counting the number of objects in a neighborhood defined by eps. Such density estimates can be highly sensitive to the radius value used.
  - Clustering method based on a set of density distribution functions.
  - Each observed object is treated as an indicator of high-probability density in the surrounding region. The probability density at a point depends on the distances from this point to the observed objects
  - Let $x_1, \ldots, x_n$ be an independent and identically distributed sample of a random variable *f*. The kernel density approximation of the probability density function is

$$\hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{\boldsymbol{x}-\boldsymbol{x_i}}{h}\right)$$

DENCLUE: Using Statistical Density Functions

Kernel K() is a non-negative real-valued integrable function that should satisfy two requirements for all values of u:

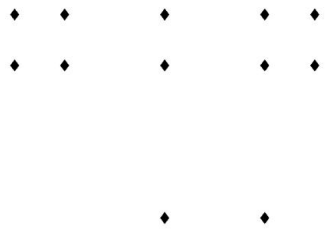
$$\int_{-\infty}^{+\infty} K(u)du = 1 \text{ and } K(-u) = K(u)$$

An example of kernel function is

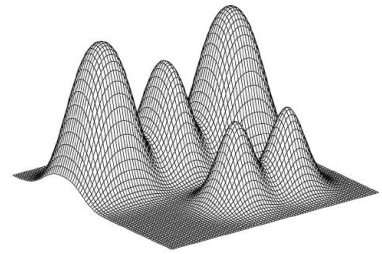
$$K\left(\frac{\boldsymbol{x}-\boldsymbol{x_i}}{h}\right) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(\boldsymbol{x}-\boldsymbol{x_i})^2}{2h^2}}$$

DENCLUE: Using Statistical Density Functions

Example of density from Gaussian Kernel



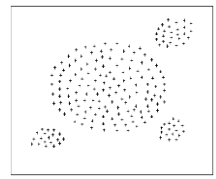
Set of 12 points.



Overall density—surface plot.

DENCLUE: Using Statistical Density Functions

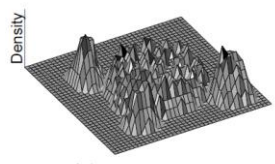
- Examples of overall density functions



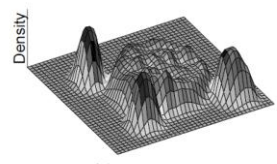
(a) Data Set

$$K_{Square}\left(\frac{x-xi}{h}\right)=\begin{cases}0 & if\ \frac{|x-xi|}{h}>1\\ 1 & otherwise\end{cases}$$

$$K_{Gauss}\left(\frac{x-xi}{h}\right)=\frac{1}{\sqrt{2\pi}}e^{-\frac{(x-xi)^2}{h^2}}$$



(b) Square Wave



(c) Gaussian

DENCLUE: Using Statistical Density Functions

- Clusters can be determined mathematically by identifying density attractors (local maxima of the estimated density function)
  To avoid trivial local maximum points, DENCLUE uses a noise threshold ξ and only considers those density attractors x* such that

$$\hat{f}(\boldsymbol{x}^*) \geq \xi$$

- These attractors are the centers of clusters.
- Objects are assigned to clusters through density attractors using a step-wise hill-climbing procedure

DENCLUE: Using Statistical Density Functions

- Algorithm
  - **Until there exist samples in the data set,**
    - Select a sample **x**. The density attractor for **x** is computed by using the hill-climbing procedure

$$\boldsymbol{x}^0 = \boldsymbol{x} \qquad \boldsymbol{x}^{j+1} = \boldsymbol{x}^j + \delta \frac{\nabla \hat{f}(\boldsymbol{x}^j)}{|\nabla \hat{f}(\boldsymbol{x}^j)|}$$

where δ is a parameter to control the speed of convergence and

$$\nabla \hat{f}(\boldsymbol{x}) = \frac{1}{h^{d+2} n \sum_{i=1}^{n} K\left(\frac{\boldsymbol{x}-\boldsymbol{x_i}}{h}\right)(\boldsymbol{x_i}-\boldsymbol{x})}$$

d=space dimension
K is a Gaussian

- The hill-climbing procedure stops at step *k*>0 if $\hat{f}(\boldsymbol{x}^{k+1}) < \hat{f}(\boldsymbol{x}^k)$ and assigns **x** to the density attractor $\boldsymbol{x}^* = \boldsymbol{x}^k$

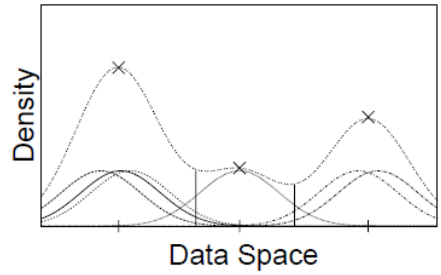
DENCLUE: Using Statistical Density Functions

- For efficiency reasons, the algorithm stores all points x' with $d(x_j, x') <= h/2$ for any step $0 < j < k$ during the hill-climbing procedure and attaches these points to the cluster of x* as well. Using this heuristics, all points which are located close to the path from **x** to its density-attractor can be classified without applying the hill-climbing procedure to them.
- An object **x** is an outlier or noise if it converges in the hill-climbing procedure to a local maximum x* with

$$\hat{f}(\boldsymbol{x}^*) < \xi$$

DENCLUE: Using Statistical Density Functions

- Examples of density attractors in a one-dimensional space



DENCLUE: Arbitrary shaped clusters

- Arbitrary shaped cluster: merge density attractors that are connected through paths of high density (> threshold)

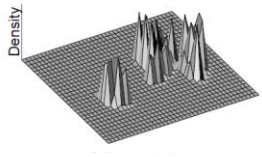
  An arbitrary-shape cluster (with respect to two constants h and ξ) for the set of density attractors X is a subset C ⊂ D, where

$$\forall x \in C \, \exists x^* \in X : \hat{f}(x^*) \geq \xi$$

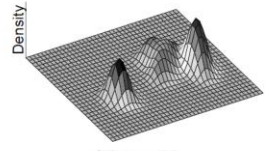
$$\forall x_1^*, x_2^* \in X : \exists \; a \; path \; P \in F^d \; from \; x_1^* \; to \; x_2^* \; with \; \forall p \in P : \hat{f}(p) \geq \xi$$

172

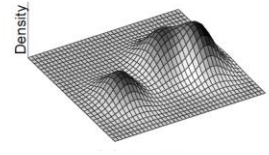
DENCLUE: Examples



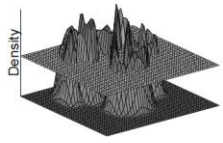
h=0.2



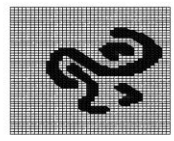
h=0.6



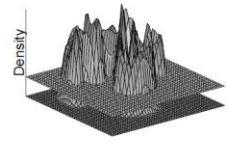
h=1.5



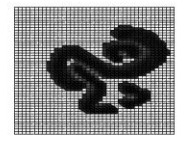
ξ=2



ξ=2



ξ=1



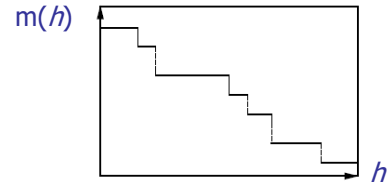
ξ=1

DENCLUE: Main Featues

- Major features
  - Solid mathematical foundation
  - Good for data sets with large amounts of noise
  - Allows a compact mathematical description of arbitrarily shaped clusters in high-dimensional data sets
  - Significant faster than existing algorithm (e.g., faster than DBSCAN by a factor of up to 45)
  - Complexity (with some optimization) O(N log(N))
- But needs an accurate choice of the parameters h and ξ.
  - h determines the influence of a point in its neighbourhood.
  - ξ describes whether a density-attractor is significant, allowing a reduction of the number of density-attractors and helping to improve how the parameters should be chosen to obtain good results

DENCLUE: Choice of the parameters

- Suggested choice for h: consider different h and determine the largest interval between h_{max} and h_{min} where the number of density attractors m(h) remains constant



- Suggested choice for ξ: if the database is noise free, all density attractors of D are significant and ξ should be chosen in

$$0 \le \xi \le \min_{x^* \in X} \{ f^{D_C}(x^*) \}$$

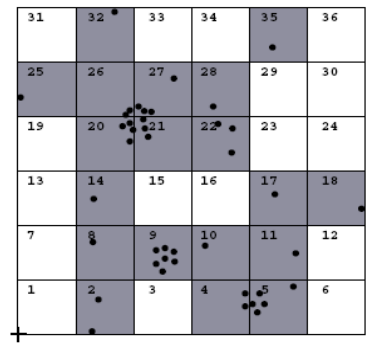
DENCLUE: Implementation

- Two steps
  - Initial pre-clustering based on a grid to speed up the calculation of the density function
  - Actual clustering
- Step 1
  - The minimal bounding (hyper-)rectangle of the data set is divided into d-dimensional hypercubes, with an edge length of 2h.
  - Only hypercubes which actually contain data points are determined.
  - The hypercubes are numbered depending on their relative position from a given origin. The keys of the populated cubes can be efficiently stored in a randomized search-tree or a B+-tree.

176

DENCLUE: Implementation

- Step 1



177

DENCLUE: Implementation

- Step 1
  - **For each populated cube c**, in addition to the key, the number of points (Nc) which belong to c, pointers to those points, and the linear sum $\sum_{x \in c} x$ are stored.
  - This information is used in the clustering step for a fast calculation of the mean of a cube (mean(c)). Since clusters can spread over more than one cube, neighboring cubes which are also populated have to be accessed.
  - To speed up this access, we connect neighboring populated cubes. More formally, two cubes c_1, $c_2 \in$ Cp are connected if d(mean(c_1), mean(c_2)) < 4σ (we use a Gaussian kernel function and therefore h=σ).

DENCLUE: Implementation

- Step 2: the clustering step
  - Only the highly populated cubes C_{sp} and cubes which are connected to a highly populated cube are considered in determining clusters.
  - This subset of Cp is denoted as

$$C_{sp} = \{c \in C_p | N_c \geq \xi_c\} \qquad C_r = C_{sp} \cup \{c \in C_p | \exists c_s \in C_{sp} \text{ and } \exists \text{connection}(c_s, c)\}$$

  - For x ∈ c and c, $c_1 \in C_r$, we set near(x) = {$x_1 \in c_1$| d(mean(c_1), x) ≤ kσ and ∃ connection(c_1, c)}.
  - The limit kσ is chosen such that only marginal influences are neglected. A value of $k = 4$ is sufficient for practical purposes
  - The resulting local density-function is

$$\hat{f}^D_{Gauss}(x) = \sum_{x_1 \in near(x)} e^{-\frac{d(x,x_1)^2}{2\sigma^2}}$$

DENCLUE: Implementation

- Step 2: the clustering step
  - The density attractor for a point x is computed as

$$x = x^0,\ x^{i+1} = x^i + \delta \frac{\nabla \hat{f}^D_{Gauss}(x^i)}{\|\nabla \hat{f}^D_{Gauss}(x^i)\|}$$

  - The computation stops at k ∈ N if

$$\hat{f}^D(x^{k+1}) < \hat{f}^D(x^k)$$

and takes

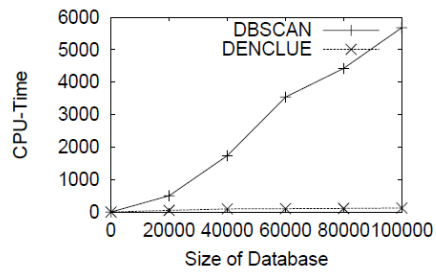
$$x^* = x^k$$

as a new density-attractor.

180

DENCLUE versus DBSCAN

- Implementation: adopts an initial pre-clustering based on a grid to speed up the calculation of the density function and a local density function.



181

Density-based clustering

- Attention
  - All the density-based clustering methods are not fully effective when clustering high dimensional data.
  - Methods that rely on near or nearest neighbor information do not work well on high dimensional spaces.
  - In high dimensional data sets, it is very unlikely that data points are nearer to each other than the average distance between data points because of sparsely filled space.
  - As a result, as the dimensionality of the space increases, the difference between the distance to the nearest and the farthest neighbors of a data object goes to zero.

182

Cluster Analysis: Basic Concepts and Methods

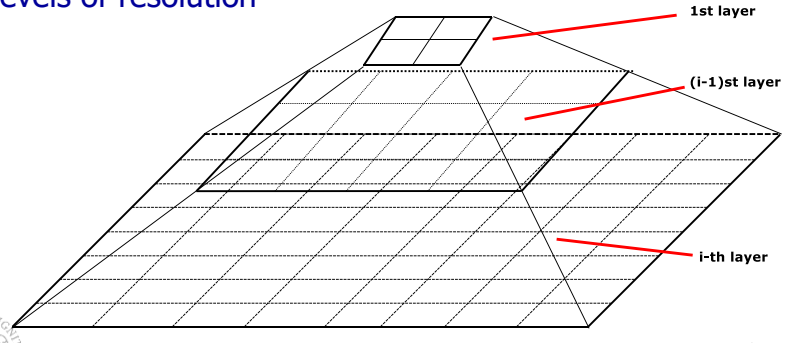
- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

Grid-Based Clustering Method

- Using multi-resolution grid data structure
- Several interesting methods
- We will analyse
  - STING (a STatistical INformation Grid approach) by Wang, Yang and Muntz (1997)
  - CLIQUE: Agrawal, et al. (SIGMOD'98)
    - Both grid-based and subspace clustering

Grid-Based Clustering Method

- Wang, Yang and Muntz (VLDB'97)
- The spatial area is divided into rectangular cells
- There are several levels of cells corresponding to different levels of resolution

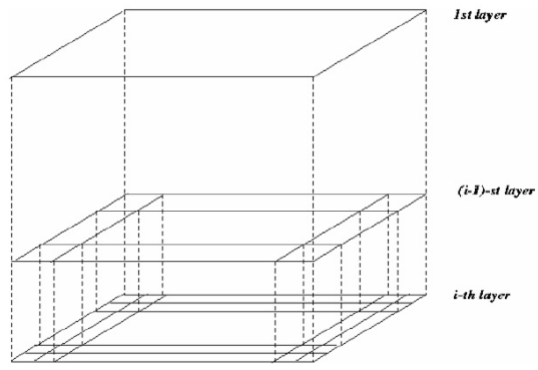


The STING Clustering Method

- Each cell at a high level is partitioned into a number of smaller cells in the next lower level
- Statistical info of each cell is calculated and stored beforehand and is used to answer queries
- Parameters of higher level cells can be easily calculated from parameters of lower level cell
- Parameters include
  - count, mean, standard deviation, min, max
  - type of distribution—normal, uniform, exponential or none (if the distribution is unknown) – obtained by the user or by hypothesis tests
- Use a top-down approach to answer spatial data queries

The STING Clustering Method

- Hierarchical structure



The STING Clustering Method

- Statistical information regarding the attributes in each grid cell, for each layer are pre-computed and stored before hand
- The statistical parameters for the cells in the lowest layer is computed directly from the values that are present in the table, when data are loaded into the database
- The statistical parameters for the cells in all the other levels are computed from their respective children cells that are in the lower level

The STING Clustering Method

- Query types
  - SQL like language used to describe queries
  - Two types of common queries found: one is to find region specifying certain constraints and other take in a region and return some attribute of the region
- Query Processing
  - We use a top-down approach to answer spatial data queries
  - Start from a pre-selected layer-typically with a small number of cells

The STING Clustering Method

- Query processing …
  - The pre-selected layer does not have to be the top most layer
  - For each cell in the current layer compute the confidence interval (or estimated range of probability) reflecting the cell's relevance to the given query
  - The confidence interval is calculated by using the statistical parameters of each cell
  - From the interval calculated we label the cells as relevant or irrelevant for this query
  - Remove the irrelevant cells from further consideration

The STING Clustering Method

- Query processing …
  - When finished with the current layer, proceed to the next lower level
  - Processing of the next lower layer examines only the remaining relevant cells
  - Repeat this process until the bottom layer is reached
  - At this time if query specifications are met, the regions of relevant cells that satisfy the query are returned
  - Otherwise, the data that fall into the relevant cells are retrieved and further processed until they meet the requirement of the query

The STING Clustering Method

- Typical query

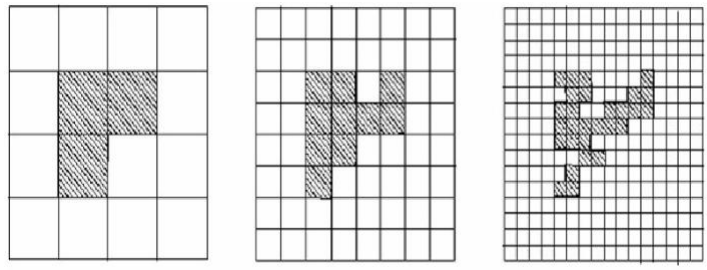
Ex1. Select the maximal regions that have at least 100 houses per unit area and at least 70% of the house prices are above \$400K and with total area at least 100 units with 90% confidence.

```
SELECT REGION
FROM house-map
WHERE DENSITY IN (100, ∞)
AND price RANGE (400000, ∞)
    WITH PERCENT (0.7, 1)
AND AREA (100, ∞)
AND WITH CONFIDENCE 0.9
```


192

The STING Clustering Method

- Typical query



Layer 1 Layer 2 Layer 3

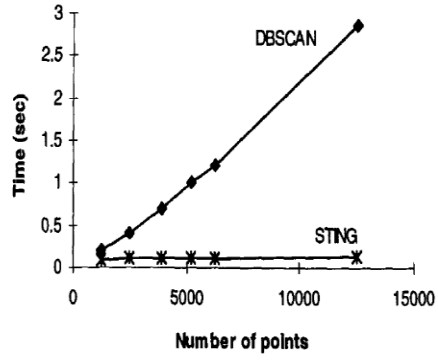
193

The STING Clustering Method

- Advantages:
  - Query-independent, easy to parallelize, incremental update
  - Generation of the clusters complexity O(n)
  - Query processing time O(g), where g is the number of grid cells at the lowest level
- Disadvantages:
  - All the cluster boundaries are either horizontal or vertical, and no diagonal boundary is detected

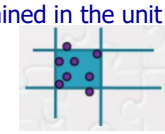
The STING Clustering Method

- The regions returned by STING are an approximation of the result by DBSCAN. As the granularity approaches zero, the regions returned by STING approach the result of DBSCAN.

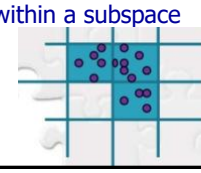


CLIQUE (Clustering In QUEst)

- Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD'98)
- Automatically identifying subspaces of a high dimensional data space that allow better clustering than original space
- CLIQUE can be considered as both density-based and grid-based
  - It partitions each dimension into the same number of equal length intervals
  - It partitions an m-dimensional data space into non-overlapping rectangular units
  - A unit is dense if the fraction of total data points contained in the unit exceeds the input model parameter



  - A cluster is a maximal set of connected dense units within a subspace



CLIQUE: The Major Steps

- Partition the data space and find the number of points that lie inside each cell of the partition.
- Identify the subspaces that contain clusters using the Apriori principle
  - If a k-dimensional unit is dense, then so are its projections in (k-1)-dimensional space. Therefore, the candidate dense units in the kth dimensional space are generated by the dense units found in (k-1)-dimensional space.
- Identify clusters
  - Determine dense units in all subspaces of interests
  - Determine connected dense units in all subspaces of interests.
- Generate minimal description for the clusters
  - Determine maximal regions that cover a cluster of connected dense units for each cluster
  - Determination of minimal cover for each cluster

CLIQUE: The algorithm

- CLIQUE performs clustering in two steps:
  - In the first step, partitions the d-dimensional data space into nonoverlapping rectangular units, identifying the dense units among these
    - CLIQUE partitions every dimension into intervals, and identifies intervals containing at least l points, where l is the density threshold.
    - CLIQUE then iteratively joins two k-dimensional dense cells, c_1 and c_2, in subspaces (Di_1 , … , Di_k) and (Dj_1 , … ,Dj_k), respectively, if $Di_1 = Dj_1$, . . . , $Di_{k-1}=Dj_{k-1}$, and c_1 and c_2 share the same intervals in those dimensions. The join operation generates a new (k+1)-dimensional candidate cell c in space (Di_1 , … , Di_k ,Dj_k).
    - CLIQUE checks whether the number of points in c passes the density threshold. The iteration terminates when no candidates can be generated or no candidate cells are dense.

198

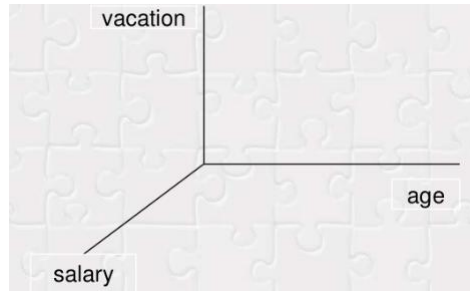
CLIQUE: The algorithm

- In the second step, CLIQUE uses the dense cells in each subspace to assemble clusters, which can be of arbitrary shape.
  - Minimum Description Length (MDL) principle: use the maximal regions to cover connected dense cells, where a maximal region is a hyperrectangle where every cell falling into this region is dense, and the region cannot be extended further in any dimension in the subspace.
  - Finding the best description of a cluster in general is NP-Hard. Thus, CLIQUE adopts a simple greedy approach. It starts with an arbitrary dense cell, finds a maximal region covering the cell, and then works on the remaining dense cells that have not yet been covered.
  - The greedy method terminates when all dense cells are covered.

199

CLIQUE: Example

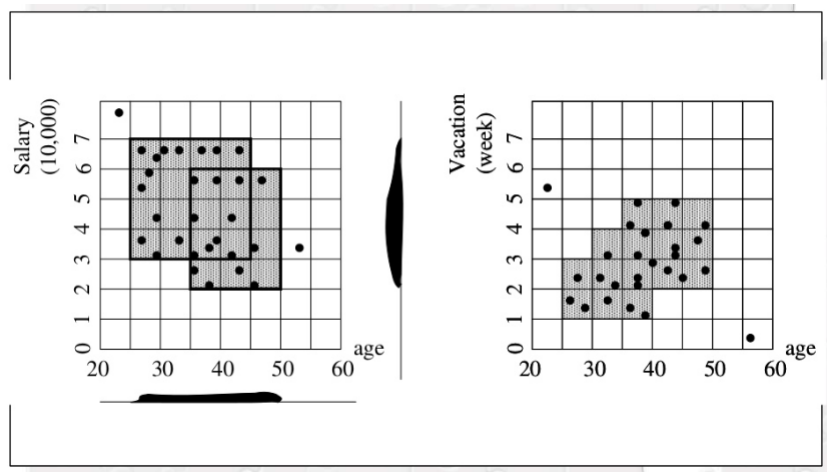
- Let us say that we want to cluster a set of records that have three attributes, namely salary, vacation and age
- The data space for this data would be 3-dimensional



CLIQUE: Example

- After plotting the data objects, each dimension (i.e., salary, vacation and age) is split into intervals of equal length
- Then, we form a 3-dimensional grid on the space, each unit of which would be a 3-D rectangle
- Now, our goal is to find the dense 3-D rectangular units
- To do this, we find the dense units of the subspaces of this 3-d space
- So, we find the dense units with respect to age for salary. This means that we look at the salary-age plane and find all the 2-D rectangular units that are dense
- We also find the dense 2-D rectangular units for the vacation-age plane

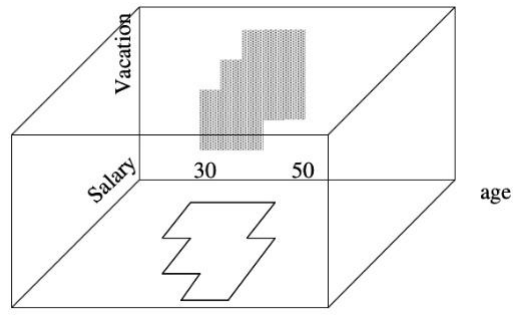
CLIQUE: Example



202

CLIQUE: Example

- Now let us try to visualize the dense units of the two planes on the following 3-d figure:



203

CLIQUE: Example

- We can extend the dense areas in the vacation-age plane inwards
- We can extend the dense areas in the salary-age plane upwards
- The intersection of these two spaces would give us as candidate search space in which 3-dimensional dense units exist
- We then find the dense units in the salary-vacation plane and we form an extension of the subspace that represents these dense units

CLIQUE: Example

- Now, we perform an intersection of the candidate search space with the extension of the dense units of the salary-vacation plane, in order to get all the 3-d dense units
- So, what was the main idea?
  - We used the dense units in subspace in order to find the dense units in the 3-dimensional space
  - After finding the dense units, it is very easy to find clusters

Strength and Weakness of *CLIQUE*

- Strength
  - automatically finds subspaces of the highest dimensionality such that high density clusters exist in those subspaces
  - insensitive to the order of records in input and does not presume some canonical data distribution
  - scales linearly with the size of input and has good scalability as the number of dimensions in the data increases
- Weakness
  - Obtaining a meaningful clustering is dependent on proper tuning of the grid size and the density threshold
  - The accuracy of the clustering result may be degraded at the expense of simplicity of the method

Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

Clustering Evaluation

- Assessing clustering tendency.
  - Clustering analysis on a data set is meaningful only when there is a nonrandom structure in the data.
- Determining the number of clusters in a dataset
- Measuring clustering quality
  - Measures to assess how well the clusters fit the dataset
  - Measures that score clustering and thus can compare two sets of clustering results on the same dataset

Determining the Number of Clusters

- Elbow method
  - increasing the number of clusters can help to reduce the sum of within-cluster variance of each cluster.
  - the marginal effect of reducing the sum of within-cluster variances may drop if too many clusters are formed, because splitting a cohesive cluster into two gives only a small reduction.
  - Use the turning point in the curve of sum of within cluster variance w.r.t the # of clusters

Determining the Number of Clusters

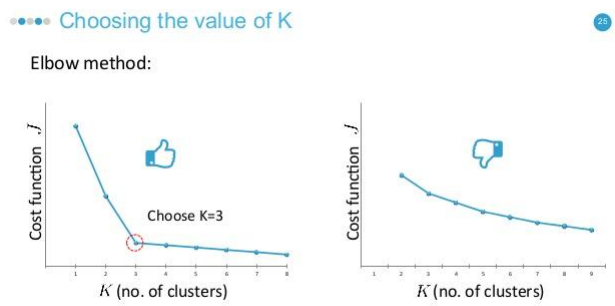
- Elbow method

Technically, given a number, k > 0,

- Form k clusters on the data set in question using a clustering algorithm like k-means, and
- Calculate the sum of within-cluster variances, var(k).
- Plot the curve of var with respect to k.
- Choose *k* as the first or most significant turning point of the curve

Determining the Number of Clusters

- Elbow method



Determine the Number of Clusters

- Cross validation method
  - Divide a given data set into m parts
  - Use m – 1 parts to obtain a clustering model
  - Use the remaining part to test the quality of the clustering
    - E.g., For each point in the test set, find the closest centroid, and use the sum of squared distance between all points in the test set and the closest centroids to measure how well the model fits the test set
- For any k > 0, repeat it m times, compare the overall quality measure w.r.t. different k's, and find # of clusters that fits the data the best

Measuring Clustering Quality

- Two methods: extrinsic vs. intrinsic
- Extrinsic: supervised, i.e., the ground truth is available
  - Compare a clustering against the ground truth using certain clustering quality measure
  - Ex. BCubed precision and recall metrics
- Intrinsic: unsupervised, i.e., the ground truth is unavailable
  - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are
  - Ex. Silhouette coefficient

Measuring Clustering Quality: Extrinsic Methods

- Clustering quality measure: Q(C, Cg), for a clustering C given the ground truth Cg.
- Q is good if it satisfies the following 4 essential criteria
  - Cluster homogeneity: the purer, the better
  - Cluster completeness: should assign objects belong to the same category in the ground truth to the same cluster
  - Rag bag: putting a heterogeneous object into a pure cluster should be penalized more than putting it into a rag bag (i.e., "miscellaneous" or "other" category)
  - Small cluster preservation: splitting a small category into pieces is more harmful than splitting a large category into pieces

Measuring Clustering Quality: Extrinsic Methods

- Bcubed precision and recall
  - Evaluates the precision and recall for every object in a clustering on a given data set according to ground truth.
  - The precision of an object indicates how many other objects in the same cluster belong to the same category as the object.
  - The recall of an object reflects how many objects of the same category are assigned to the same cluster

Formally, let $D = \{o_1, \ldots, o_n\}$ be a set of objects, and $\mathcal{C}$ be a clustering on D. Let $L(o_i)$ $(1 \leq i \leq n)$ be the category of o_i given by ground truth, and $C(o_i)$ be the *cluster_ID* of o_i in $\mathcal{C}$. Then, for two objects, o_i and o_j, $(1 \leq i, j, \leq n, i \neq j)$, the *correctness* of the relation between o_i and o_j in clustering $\mathcal{C}$ is given by

$$\text{Correctness}(o_i, o_j) = \begin{cases} 1 & \text{if } L(o_i) = L(o_j) \Leftrightarrow C(o_i) = C(o_j) \\ 0 & \text{otherwise.} \end{cases}$$

Measuring Clustering Quality: Extrinsic Methods

- Bcubed precision and recall

$$\text{Precision BCubed} = \frac{\sum_{i=1}^{n} \frac{\sum_{o_j: i \neq j, C(o_i)=C(o_j)} \text{Correctness}(o_i, o_j)}{\|\{o_j | i \neq j, C(o_i) = C(o_j)\}\|}}{n}$$

$$\text{Recall BCubed} = \frac{\sum_{i=1}^{n} \frac{\sum_{o_j: i \neq j, L(o_i)=L(o_j)} \text{Correctness}(o_i, o_j)}{\|\{o_j | i \neq j, L(o_i) = L(o_j)\}\|}}{n}$$

Measuring Clustering Quality: Intrinsic Methods

- When the ground truth is not available, we have to use an intrinsic method to assess the clustering quality.
- Intrinsic methods evaluate a clustering by examining how well the clusters are separated and how compact the clusters are.
- Silhouette coefficient
  - For each object **o** in *D*,
    - compute a(**o**) as the average distance between **o** and all the other objects in the cluster to which **o** belongs to.
    - Compute b(**o**) as the minimum average distance from **o** to all clusters to which **o** does not belong to.

Measuring Clustering Quality: Intrinsic Methods

$$a(o) = \frac{\sum_{o' \in C_i, o \neq o'} dist(o, o')}{|C_i| - 1}$$

$$b(o) = \min_{C_j : 1 \leq j \leq k, j \neq i} \left\{ \frac{\sum_{o' \in C_j} dist(o, o')}{|C_j|} \right\}$$

- Silhouette coefficient of **o**

$$s(o) = \frac{b(o) - a(o)}{\max\{a(o), b(o)\}}$$

Measuring Clustering Quality: Intrinsic Methods

- The value of the silhouette coefficient is between -1 and 1. The smaller the value of a(o), the more compact the cluster.
- When s(**o**) approaches 1, the cluster containing **o** is compact and **o** is far away from other clusters, which is the preferable case.
- When the silhouette coefficient value is negative (i.e., b(**o**) < a(**o**)), this means that, in expectation, **o** is closer to the objects in another cluster than to the objects in the same cluster as **o**. This is a bad situation and should be avoided.
- To measure a cluster's fitness within a clustering, we can compute the average silhouette coefficient value of all objects in the cluster.
- To measure the quality of a clustering, we can use the average silhouette coefficient value of all objects in the data set.

Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

Summary

- Cluster analysis groups objects based on their similarity and has wide applications
- Measure of similarity can be computed for various types of data
- Clustering algorithms can be categorized into partitioning methods, hierarchical methods, density-based methods, grid-based methods, and model-based methods
- K-means and K-medoids algorithms are popular partitioning-based clustering algorithms
- Birch and Chameleon are interesting hierarchical clustering algorithms, and there are also probabilistic hierarchical clustering algorithms
- DBSCAN, OPTICS, and DENCLU are interesting density-based algorithms
- STING and CLIQUE are grid-based methods, where CLIQUE is also a subspace clustering algorithm
- Quality of clustering results can be evaluated in various ways