

# Data Mining:

---

# Concepts and Techniques

(3<sup>rd</sup> ed.)

## — Chapter 10 —

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

©2011 Han, Kamber & Pei. All rights reserved.



# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

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

# What is 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

# Clustering for Data Understanding and 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
- Earthquake 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 (Utility)

---

- 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 (I)**

---

- Partitioning approach:
  - Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
  - Typical methods: k-means, k-medoids, CLARANS
- Hierarchical approach:
  - Create a hierarchical decomposition of the set of data (or objects) using some criterion
  - Typical methods: Diana, Agnes, BIRCH, CAMELEON
- Density-based approach:
  - Based on connectivity and density functions
  - Typical methods: DBSACN, OPTICS, DenClue
- Grid-based approach:
  - based on a multiple-level granularity structure
  - Typical methods: STING, WaveCluster, CLIQUE

# Major Clustering Approaches (II)

---

- 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

# Chapter 10. Cluster Analysis: 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  $C_i$ )

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

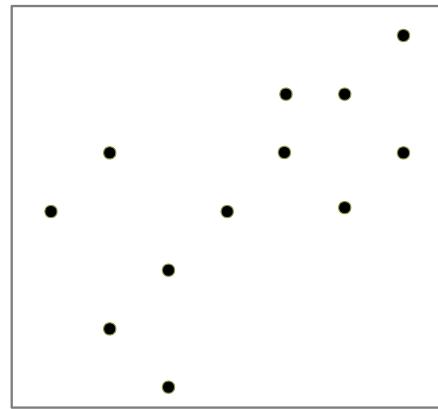
- Given  $k$ , find a partition of  $k$  *clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - *k-means* (MacQueen'67, Lloyd'57/'82): Each cluster is represented by the center 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

---

- Given  $k$ , the *k-means* algorithm is implemented in four steps:
  - Partition objects into  $k$  nonempty subsets
  - Compute seed points as the centroids of the clusters of the current partitioning (the centroid is the center, i.e., *mean point*, of the cluster)
  - Assign each object to the cluster with the nearest seed point
  - Go back to Step 2, stop when the assignment does not change

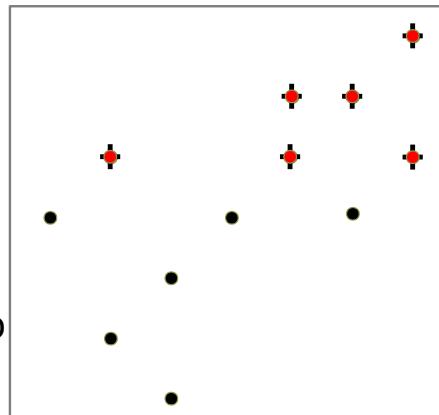
# An Example of *K-Means* Clustering



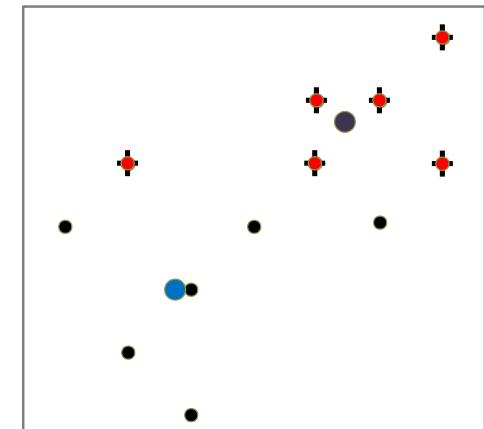
K=2

Arbitrarily partition objects into k groups

The initial data set

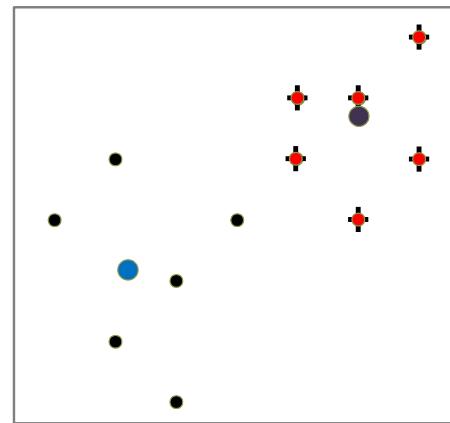


Update the cluster centroids

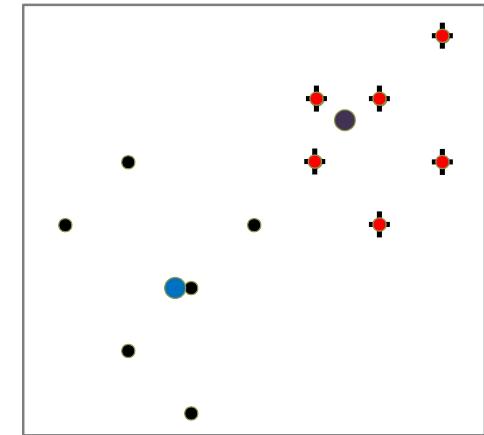


Reassign objects

Loop if needed



Update the cluster centroids



- 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

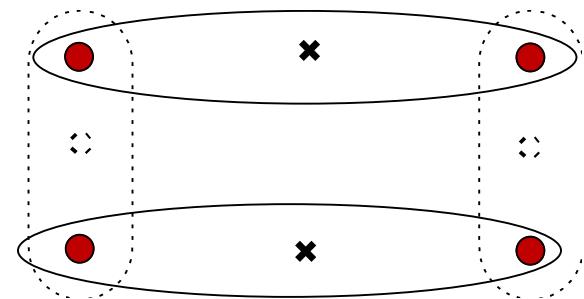
---

- Strength: *Efficient*:  $O(tkn)$ , where  $n$  is # objects,  $k$  is # clusters, and  $t$  is # iterations. Normally,  $k, t \ll n$ .
  - Comparing: PAM:  $O(k(n-k)^2)$ , CLARA:  $O(ks^2 + k(n-k))$
- 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  $k$  (see Hastie et al., 2009))
  - Sensitive to noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

# Variations of the *K-Means* Method

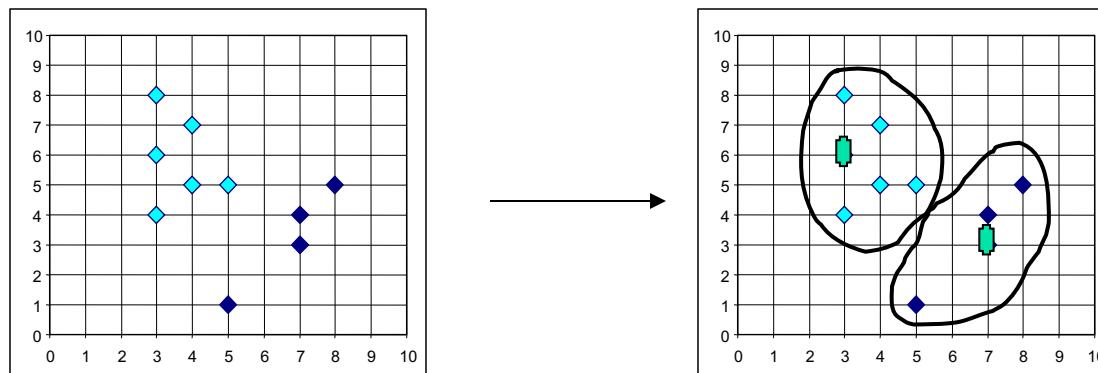
---

- Most of the variants of the *k-means* which 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
  - 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

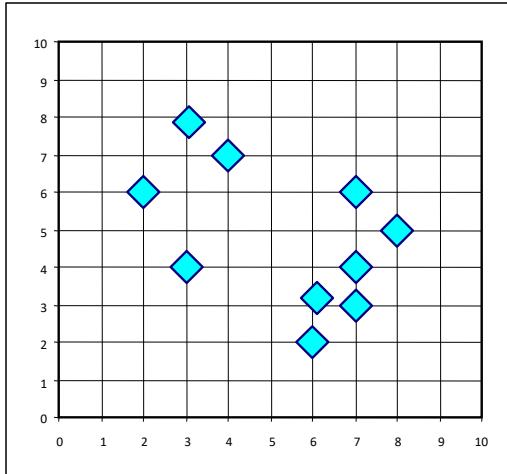


# What Is the Problem of the K-Means Method?

- 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, which is the **most centrally located** object in a cluster

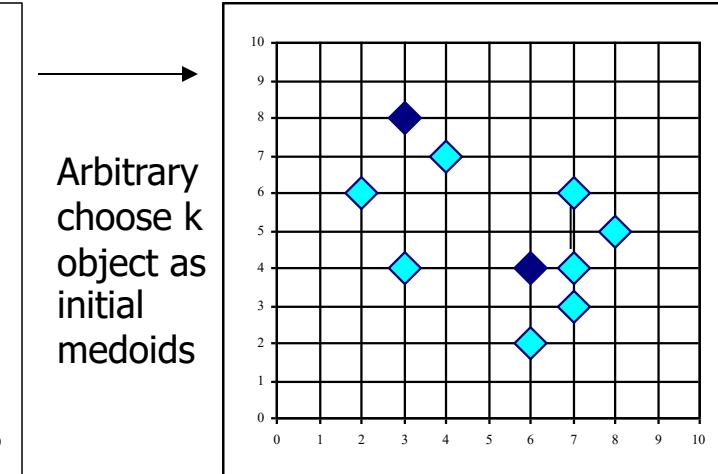


# PAM: A Typical K-Medoids Algorithm

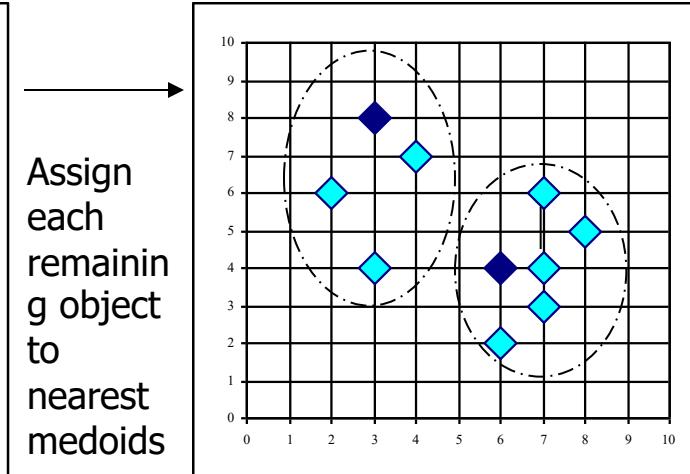


K=2

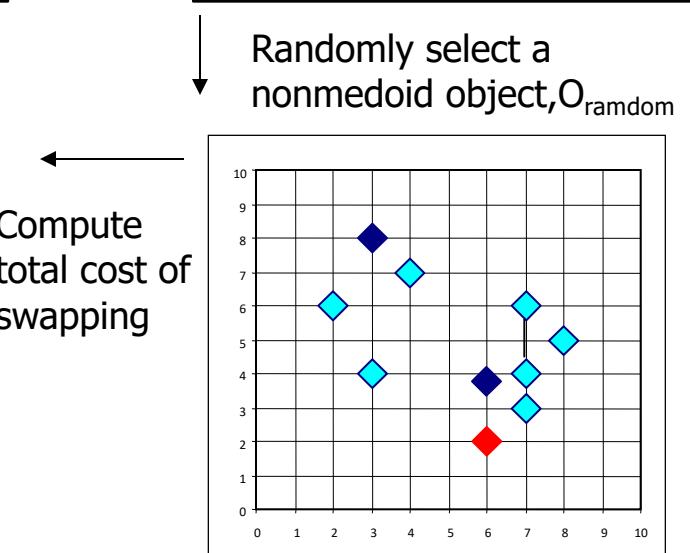
**Do loop**  
**Until no change**



Arbitrary choose k object as initial medoids

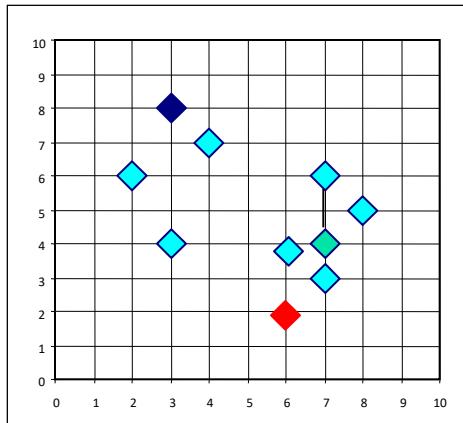


Assign each remainin g object to nearest medoids



Total Cost = 26  
Randomly select a nonmedoid object,  $O_{\text{random}}$

Compute total cost of swapping



Swapping  $O$  and  $O_{\text{random}}$   
If quality is improved.

# The K-Medoid Clustering Method

---

- *K-Medoids* Clustering: Find *representative* objects (medoids) in clusters
  - *PAM* (Partitioning Around Medoids, Kaufmann & Rousseeuw 1987)
    - Starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
    - *PAM* works effectively for small data sets, but does not scale well for large data sets (due to the computational complexity)
- Efficiency improvement on PAM
  - *CLARA* (Kaufmann & Rousseeuw, 1990): PAM on samples
  - *CLARANS* (Ng & Han, 1994): Randomized re-sampling

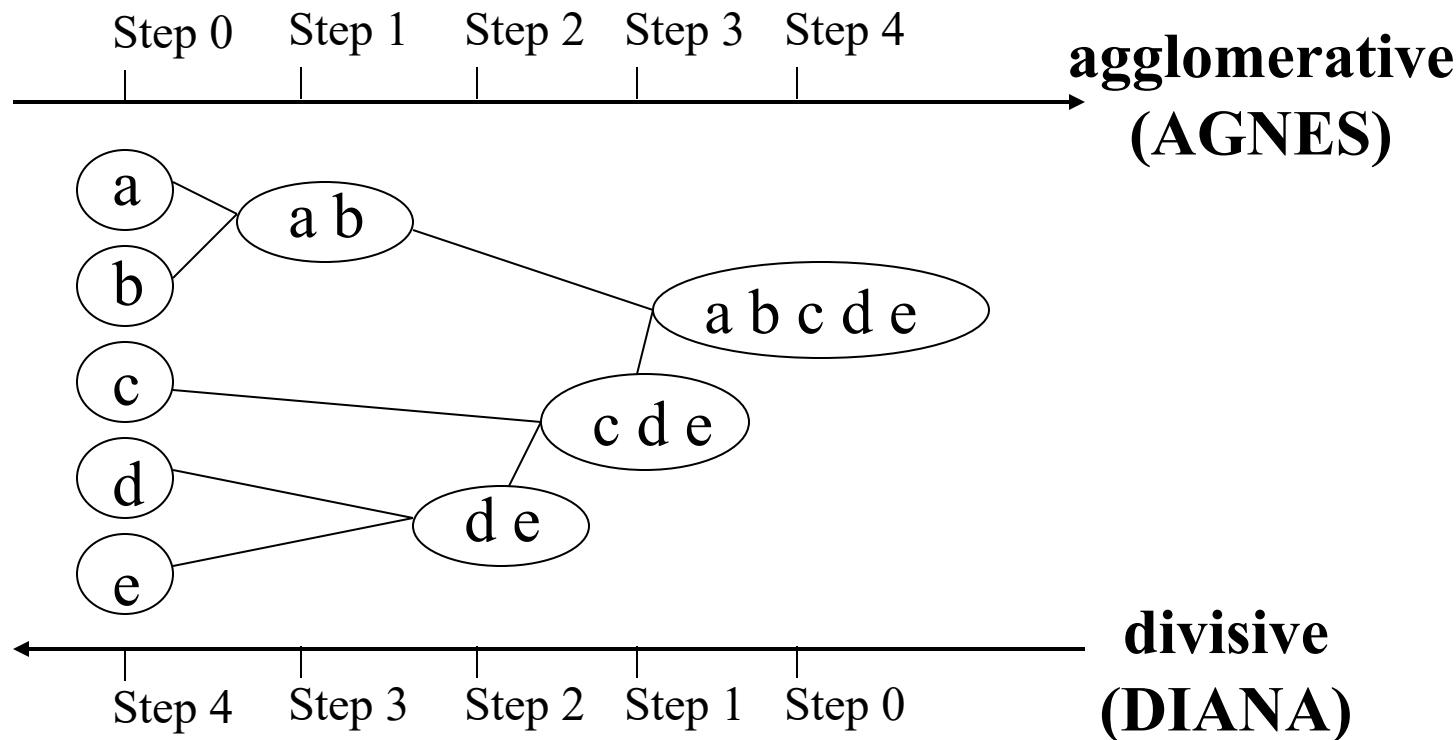
# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

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

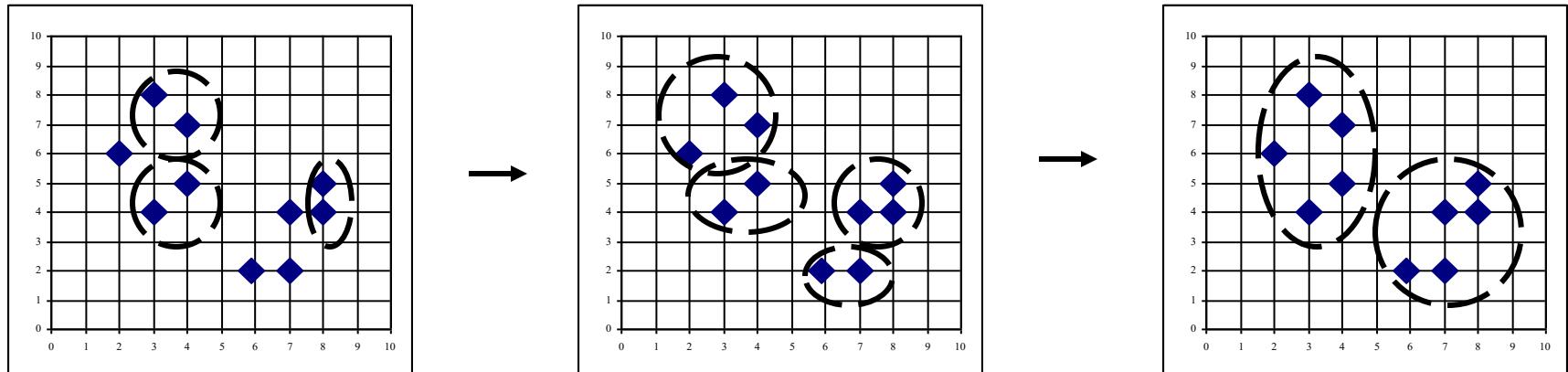
# Hierarchical Clustering

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



# AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical packages, e.g., Splus
- Use the **single-link** method and the dissimilarity matrix
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster

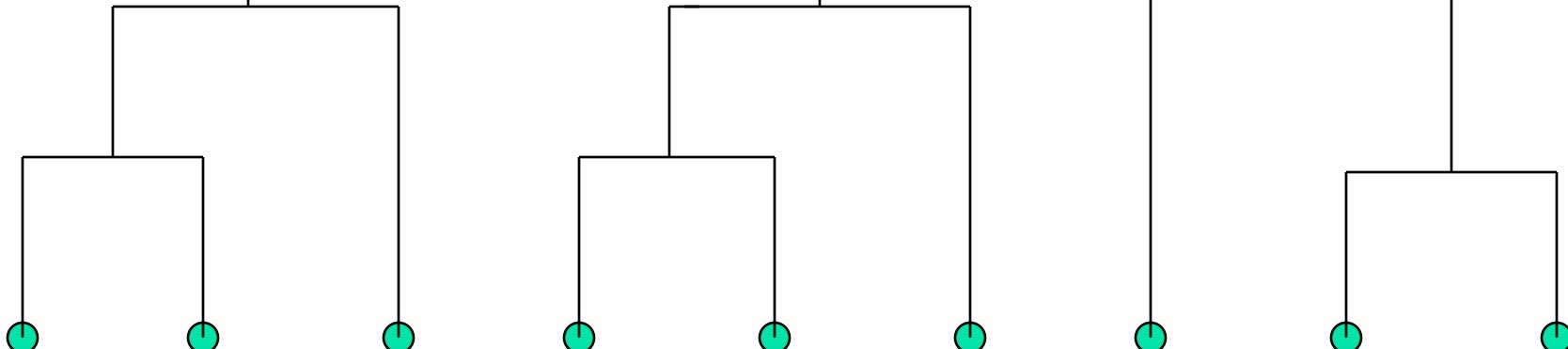


# Dendrogram: Shows How Clusters are Merged

---

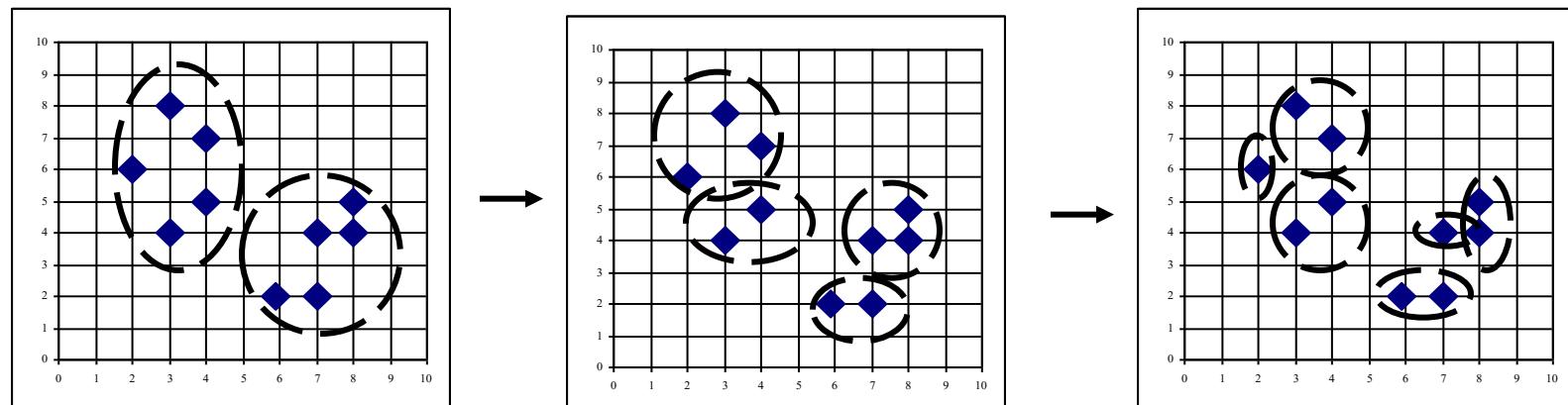
Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram

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

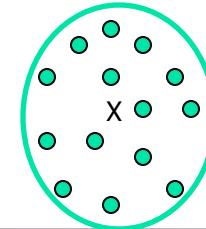


# DIANA (Divisive Analysis)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Inverse order of AGNES
- Eventually each node forms a cluster on its own



# Distance between Clusters



- Single link: smallest distance between an element in one cluster and an element in the other, i.e.,  $\text{dist}(K_i, K_j) = \min(t_{ip}, t_{jq})$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e.,  $\text{dist}(K_i, K_j) = \max(t_{ip}, t_{jq})$
- Average: avg distance between an element in one cluster and an element in the other, i.e.,  $\text{dist}(K_i, K_j) = \text{avg}(t_{ip}, t_{jq})$
- Centroid: distance between the centroids of two clusters, i.e.,  $\text{dist}(K_i, K_j) = \text{dist}(C_i, C_j)$
- Medoid: distance between the medoids of two clusters, i.e.,  $\text{dist}(K_i, K_j) = \text{dist}(M_i, M_j)$ 
  - Medoid: a chosen, centrally located object in the cluster

# Centroid, Radius and Diameter of a Cluster (for numerical data sets)

---

- Centroid: the “middle” of a cluster
- Radius: square root of average distance from any point of the cluster to its centroid
- Diameter: square root of average mean squared distance between all pairs of points in the cluster

$$C_m = \frac{\sum_{i=1}^N (t_{ip})}{N}$$

$$R_m = \sqrt{\frac{\sum_{i=1}^N (t_{ip} - c_m)^2}{N}}$$

$$D_m = \sqrt{\frac{\sum_{i=1}^N \sum_{j=1}^N (t_{ip} - t_{iq})^2}{N(N-1)}}$$

# Extensions to Hierarchical Clustering

---

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

# BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies)

---

- Zhang, Ramakrishnan & Livny, SIGMOD'96
- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering
  - Phase 1: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)
  - Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree
- *Scales linearly*: finds a good clustering with a single scan and improves the quality with a few additional scans
- Weakness: handles only numeric data, and sensitive to the order of the data record

# Clustering Feature Vector in BIRCH

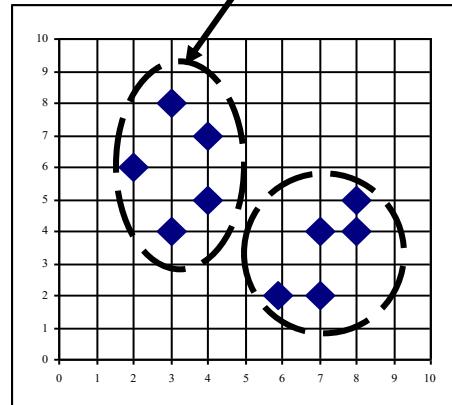
Clustering Feature (CF):  $CF = (N, LS, SS)$

$N$ : Number of data points

$LS$ : linear sum of  $N$  points:  $\sum_{i=1}^N X_i$

$SS$ : square sum of  $N$  points

$$\sum_{i=1}^N X_i^2$$



$$CF = (5, (16,30),(54,190))$$

(3,4)

(2,6)

(4,5)

(4,7)

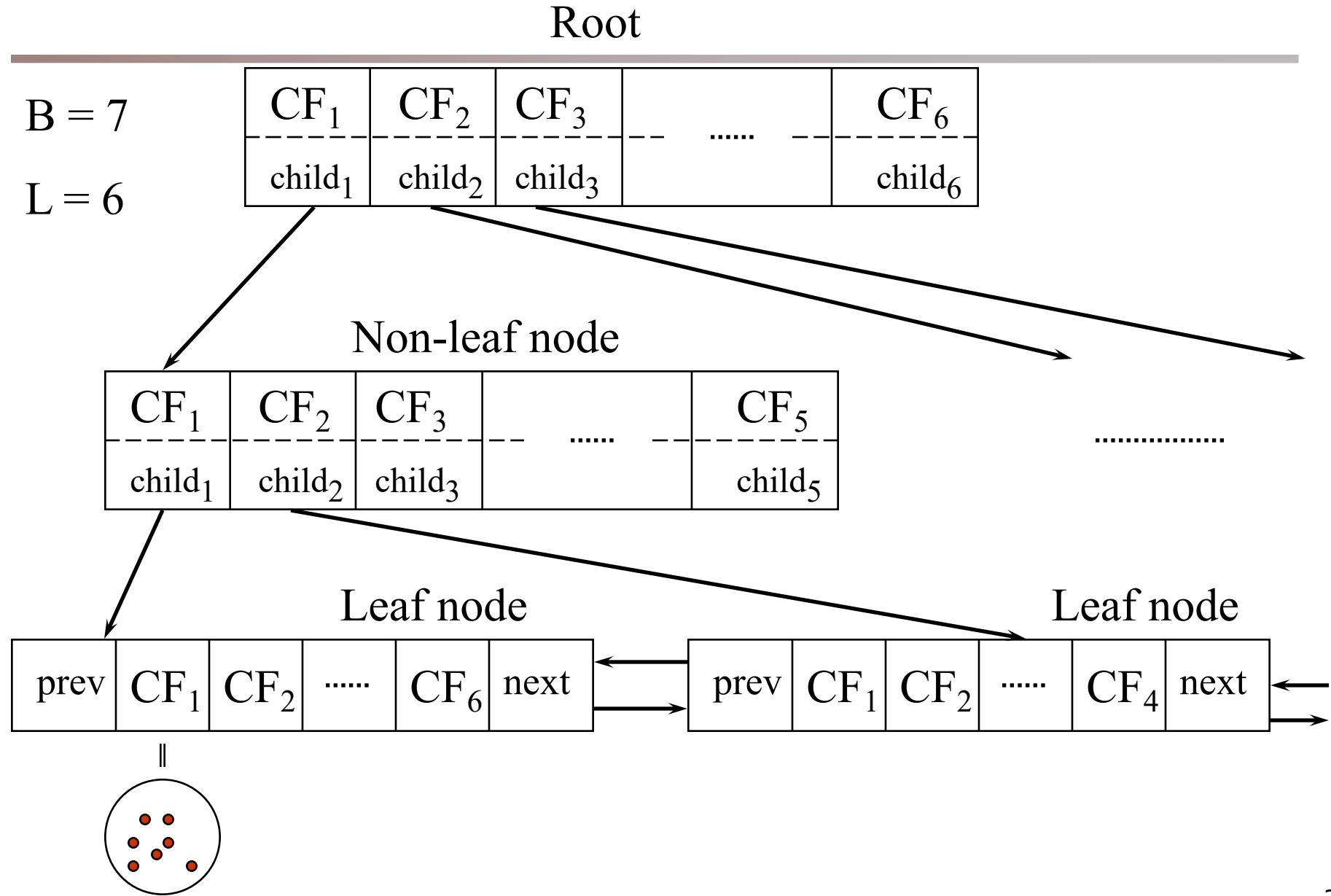
(3,8)

# CF-Tree in BIRCH

---

- Clustering feature:
  - Summary of the statistics for a given subcluster: the 0-th, 1st, and 2nd moments of the subcluster from the statistical point of view
  - Registers crucial measurements for computing cluster and utilizes storage efficiently
- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering
  - A nonleaf node in a tree has descendants or “children”
  - The nonleaf nodes store sums of the CFs of their children
- A CF tree has two parameters
  - Branching factor: max # of children
  - Threshold: max diameter of sub-clusters stored at the leaf nodes

# The CF Tree Structure



# The Birch Algorithm

---

- Cluster Diameter

$$\sqrt{\frac{1}{n(n-1)} \sum (x_i - x_j)^2}$$

- For each point in the input
  - Find closest leaf entry
  - Add point to leaf entry and update CF
  - If entry diameter > max\_diameter, then split leaf, and possibly parents
- Algorithm is O(n)
- Concerns
  - 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

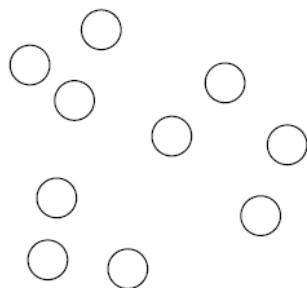
# CHAMELEON: Hierarchical Clustering Using Dynamic Modeling (1999)

---

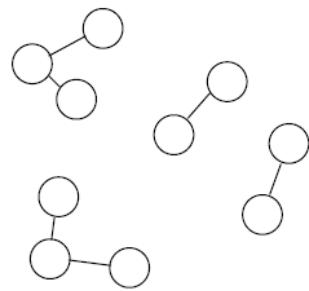
- CHAMELEON: G. Karypis, E. H. Han, and V. Kumar, 1999
- Measures the similarity based on a dynamic model
  - Two clusters are merged only if the *interconnectivity* and *closeness (proximity)* between two clusters are high *relative to* the internal interconnectivity of the clusters and closeness of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

# k-nearest neighbor graph

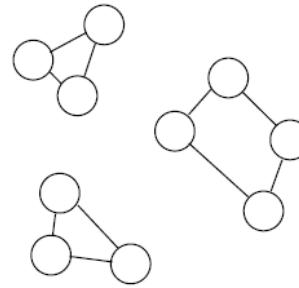
- *k-nearest graphs from an original data in 2D*



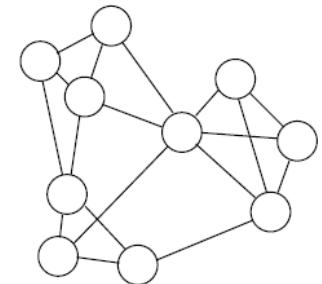
(a) Original Data in 2D



(b) 1-nearest neighbor graph



(c) 2-nearest neighbor graph



(d) 3-nearest neighbor graph

# Inter-connectivity

---

- $EC_{\{C_i, C_j\}}$  The absolute inter-connectivity between a pair of clusters  $C_i$  and  $C_j$  is defined to be as the sum of the weight of the edges that connect vertices in  $C_i$  to vertices in  $C_j$ .
- The internal inter-connectivity of a cluster  $C_i$  can be easily captured by the size of its min-cut bisector  $EC_{C_i}$  (i.e., the weighted sum of edges that partition the graph into two roughly equal parts).
- Relative Inter-connectivity (RI)

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

# Relative Closeness

---

- The **relative closeness** between a pair of clusters  $C_i$  and  $C_j$  is defined as the absolute closeness between  $C_i$  and  $C_j$  normalized with respect to the internal closeness of the two clusters  $C_i$  and  $C_j$ .

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

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

# Merge Sub-Clusters

---

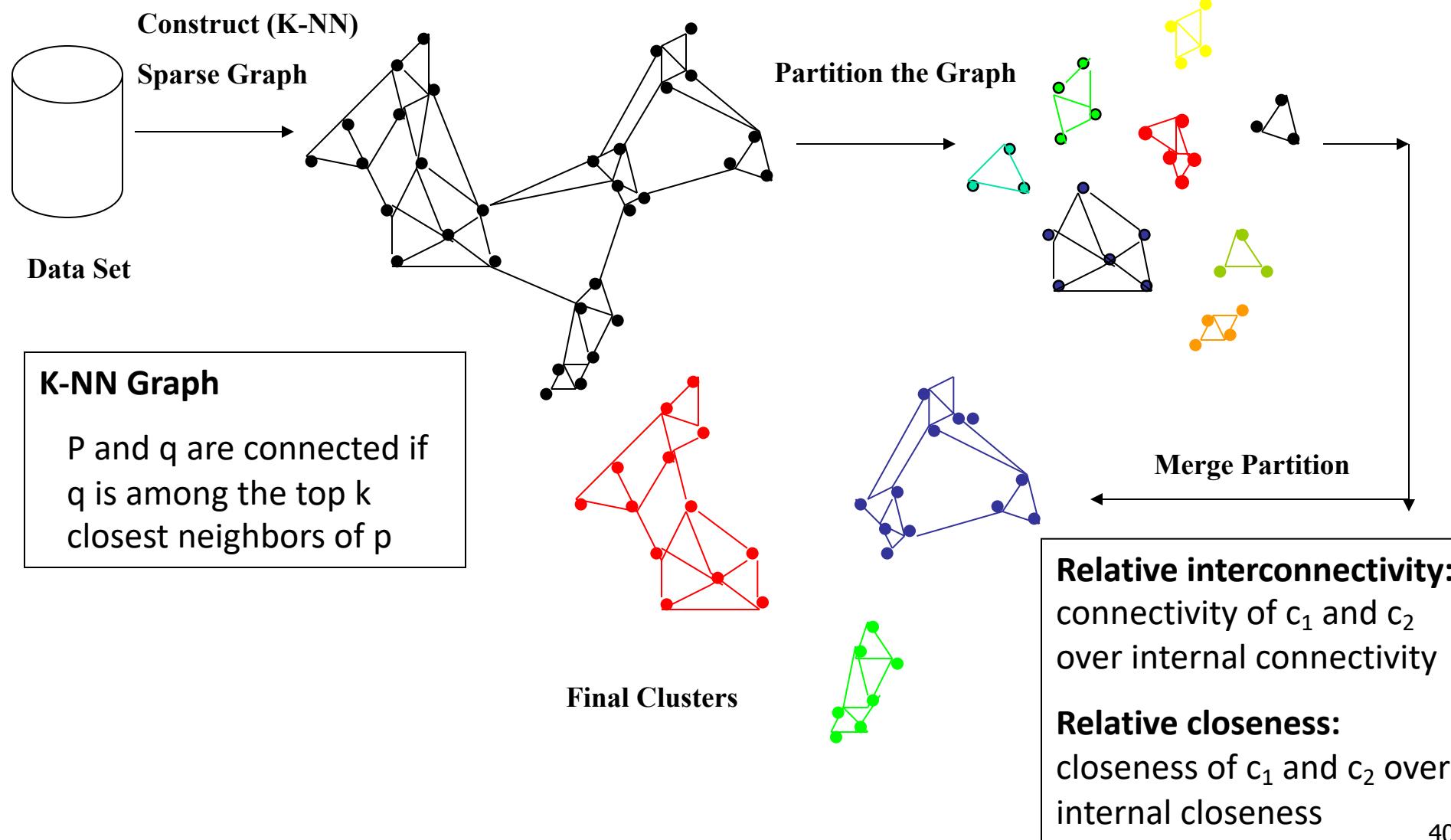
- The **first scheme** merges only those pairs of clusters whose relative inter-connectivity and relative closeness are both above some user specified threshold  $T_{RI}$  and  $T_{RC}$ , respectively.

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

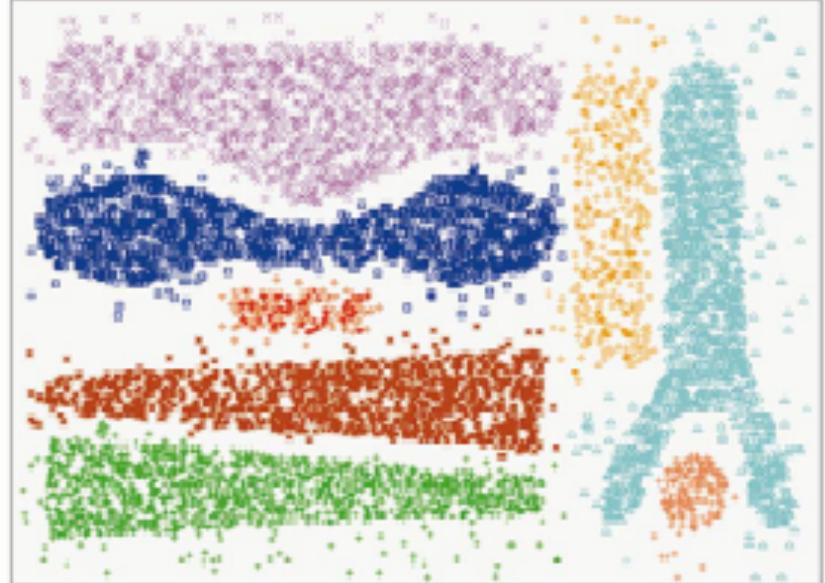
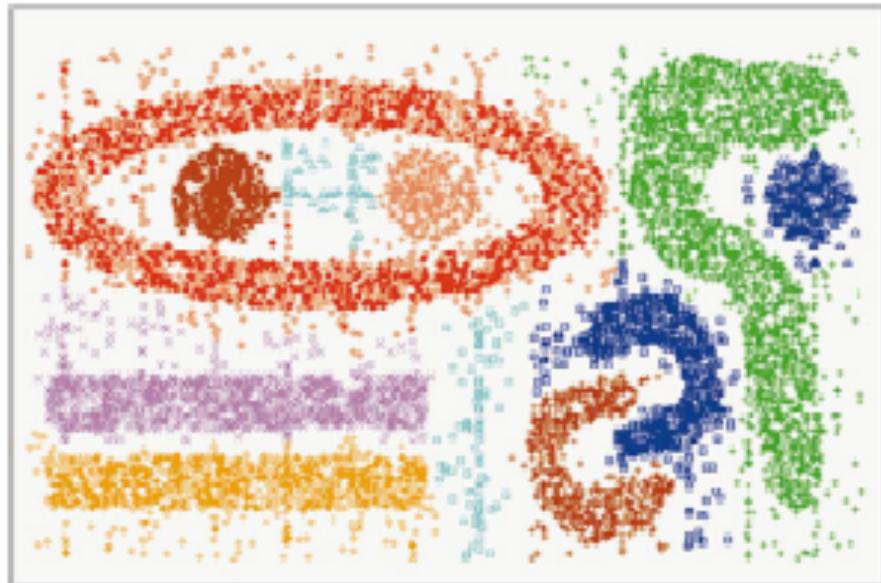
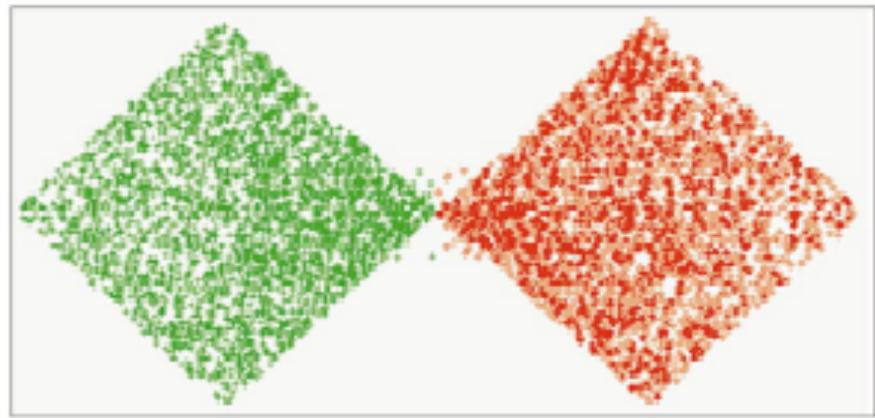
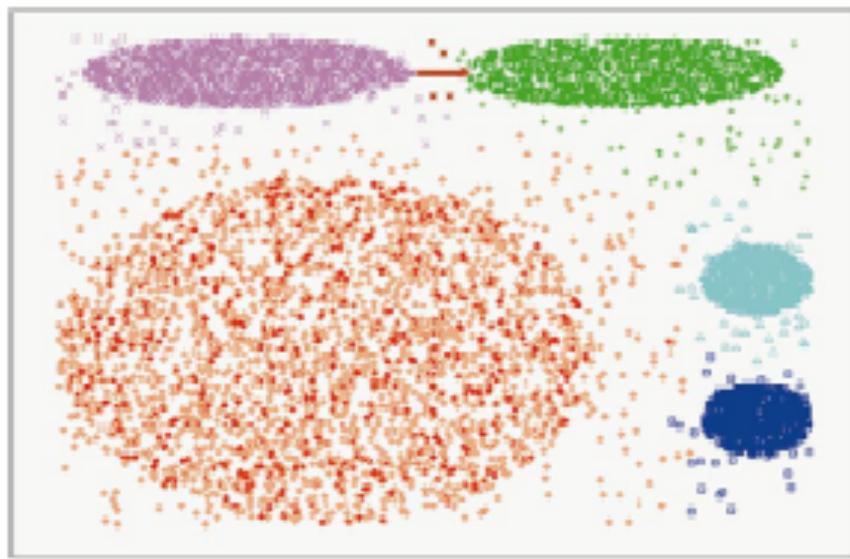
- The **second scheme** uses a function to combine the relative inter-connectivity and relative closeness, and then selects to merge the pair of clusters that maximizes this function.

$$RI(C_i, C_j) * RC(C_i, C_j)^\alpha$$

# Overall Framework of CHAMELEON



# CHAMELEON (Clustering Complex Objects)



# Why called CHAMELEON?

---



# Probabilistic Hierarchical Clustering

---

- Algorithmic hierarchical clustering
  - Nontrivial to choose a good distance measure
  - Hard to handle missing attribute values
  - Optimization goal not clear: heuristic, local search
- Probabilistic hierarchical clustering
  - Use probabilistic models to measure distances between clusters
  - Generative model: Regard the set of data objects to be clustered as a sample of the underlying data generation mechanism to be analyzed
  - Easy to understand, same efficiency as algorithmic agglomerative clustering method, can handle partially observed data
- In practice, assume the generative models adopt common distributions functions, e.g., Gaussian distribution or Bernoulli distribution, governed by parameters

# Generative Model

---

- Given a set of 1-D points  $X = \{x_1, \dots, x_n\}$  for clustering analysis & assuming they are generated by a Gaussian distribution:
- The probability that a point  $x_i \in X$  is generated by the model
- The likelihood that  $X$  is generated by the model:

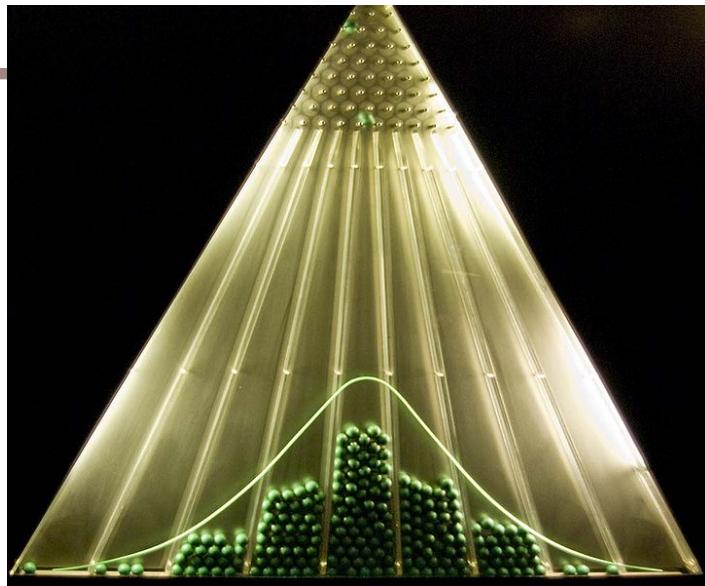
$$L(\mathcal{N}(\mu, \sigma^2) : X) = P(X|\mu, \sigma^2) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}}$$

- The task of learning the generative model: find the parameters  $\mu$  and  $\sigma^2$  such that

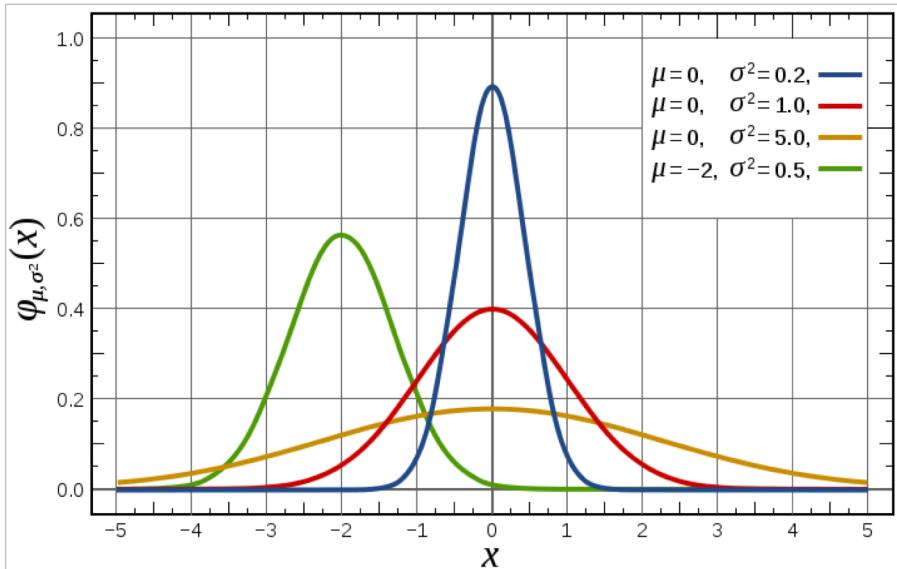
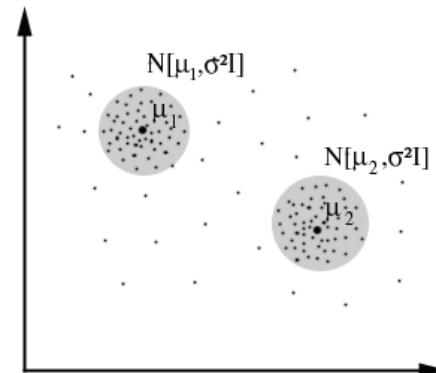
the maximum likelihood



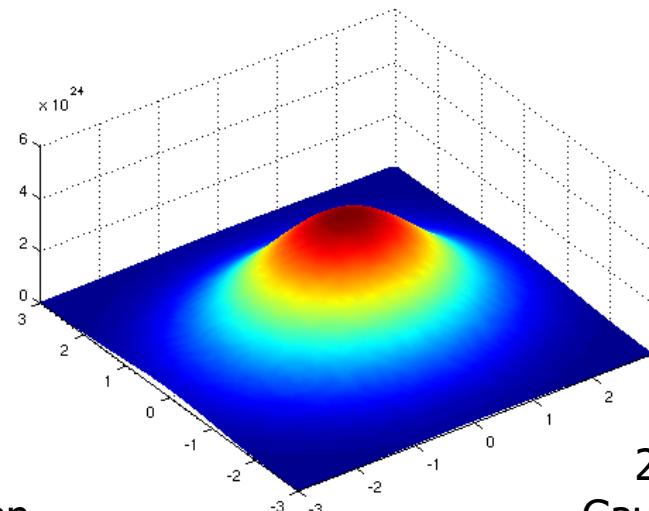
# Gaussian Distribution



Bean  
machine:  
drop ball  
with pins



1-d  
Gaussian



2-d  
Gaussian

From wikipedia and <http://home.dei.polimi.it>

# A Probabilistic Hierarchical Clustering Algorithm

- For a set of objects partitioned into  $m$  clusters  $C_1, \dots, C_m$ , the quality can be measured by,

$$Q(\{C_1, \dots, C_m\}) = \prod_{i=1}^m P(C_i)$$

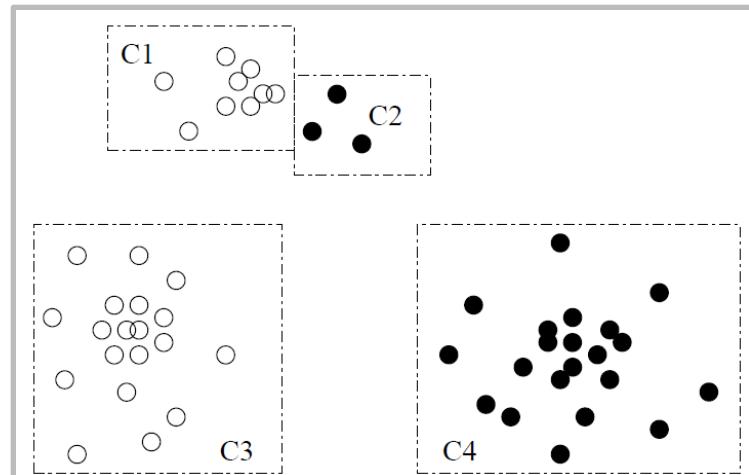
where  $P()$  is the maximum likelihood

- If we merge two clusters  $C_{j_1}$  and  $C_{j_2}$  into a cluster  $C_{j_1} \cup C_{j_2}$ , then, the change in quality of the overall clustering is

$$\begin{aligned} & Q((\{C_1, \dots, C_m\} - \{C_{j_1}, C_{j_2}\}) \cup \{C_{j_1} \cup C_{j_2}\}) - Q(\{C_1, \dots, C_m\}) \\ = & \frac{\prod_{i=1}^m P(C_i) \cdot P(C_{j_1} \cup C_{j_2})}{P(C_{j_1})P(C_{j_2})} - \prod_{i=1}^m P(C_i) \\ = & \prod_{i=1}^m P(C_i) \left( \frac{P(C_{j_1} \cup C_{j_2})}{P(C_{j_1})P(C_{j_2})} - 1 \right) \end{aligned}$$

- Distance between clusters  $C_1$  and  $C_2$ :

$$dist(C_i, C_j) = -\log \frac{P(C_1 \cup C_2)}{P(C_1)P(C_2)}$$



- 
- Algorithm: Progressively merge points and clusters

Input:  $D = \{o_1, \dots, o_n\}$ : a data set containing n objects

Output: A hierarchy of clusters

Method

Create a cluster for each object  $C_i = \{o_i\}$ ,  $1 \leq i \leq n$ ;

For  $i = 1$  to  $n$  {

Find pair of clusters  $C_i$  and  $C_j$  such that

$$C_i, C_j = \operatorname{argmax}_{i \neq j} \{\log (P(C_i \cup C_j) / (P(C_i)P(C_j)))\};$$

If  $\log (P(C_i \cup C_j) / (P(C_i)P(C_j))) > 0$  then merge  $C_i$  and  $C_j$

}

# Chapter 10. 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

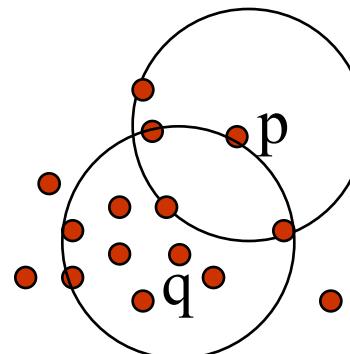
---

- 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
- $N_{Eps}(q)$ : {p belongs to D |  $\text{dist}(p,q) \leq Eps$ }
- **Directly density-reachable**: A point  $p$  is directly density-reachable from a point  $q$  w.r.t. *Eps*, *MinPts* if
  - $p$  belongs to  $N_{Eps}(q)$
  - core point condition:

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



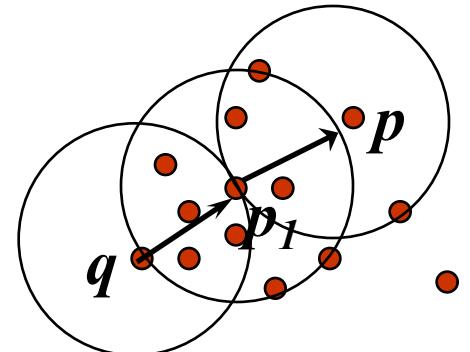
$\text{MinPts} = 5$

$Eps = 1 \text{ cm}$

# Density-Reachable and Density-Connected

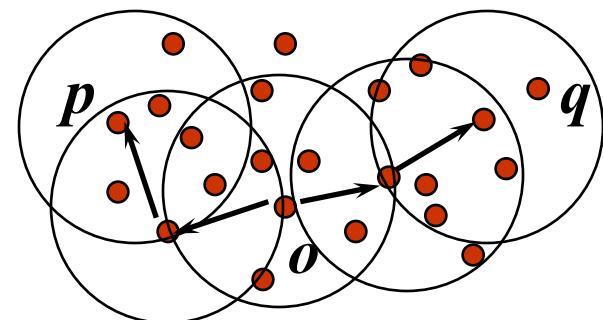
- Density-reachable:

- A point  $p$  is **density-reachable** from a point  $q$  w.r.t.  $Eps$ ,  $MinPts$  if there is a chain of points  $p_1, \dots, p_n$ ,  $p_1 = q$ ,  $p_n = p$  such that  $p_{i+1}$  is directly density-reachable from  $p_i$ .



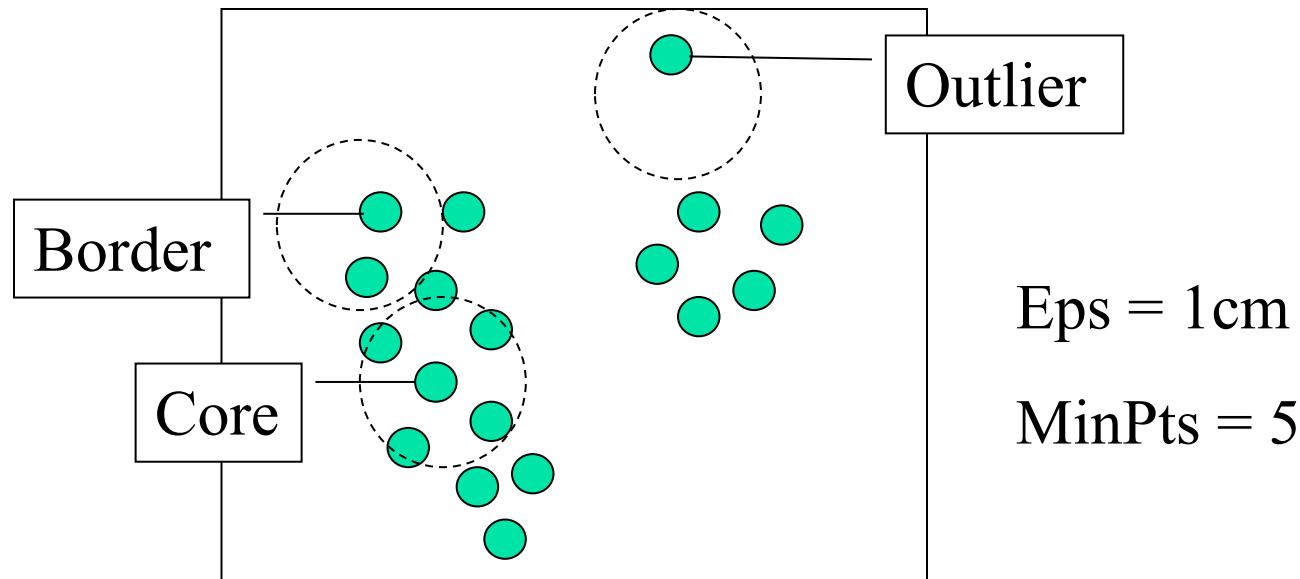
- Density-connected

- A point  $p$  is **density-connected** to a point  $q$  w.r.t.  $Eps$ ,  $MinPts$  if there is a point  $o$  such that both,  $p$  and  $q$  are density-reachable from  $o$  w.r.t.  $Eps$  and  $MinPts$ .



# DBSCAN: Density-Based Spatial Clustering of Applications with Noise

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



# DBSCAN: The Algorithm

---

- Arbitrary select a point  $p$
- Retrieve all points density-reachable from  $p$  w.r.t.  $Eps$  and  $MinPts$
- If  $p$  is a core point, a cluster is formed
- If  $p$  is a border point, no points are density-reachable from  $p$  and DBSCAN visits the next point of the database
- Continue the process until all of the points have been processed
- *If a spatial index is used, the computational complexity of DBSCAN is  $O(n \log n)$ , where  $n$  is the number of database objects. Otherwise, the complexity is  $O(n^2)$*

# DBSCAN: Sensitive to Parameters

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

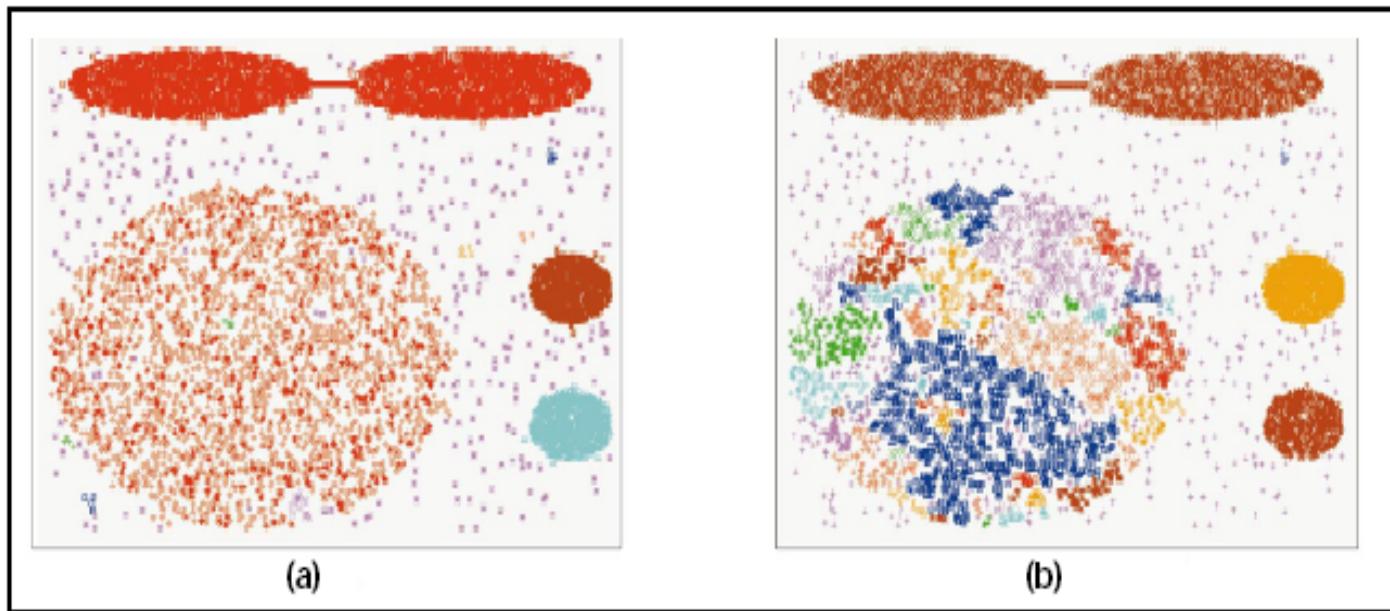
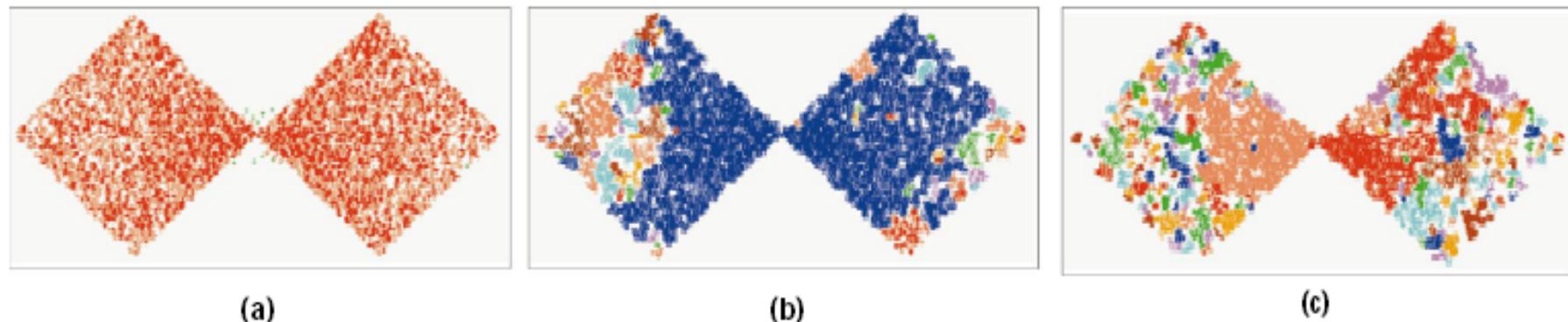


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



# DBSCAN online Demo

---

- <http://webdocs.cs.ualberta.ca/~yaling/Cluster/Applet/Code/Cluster.html>

# 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 wrt its density-based clustering structure
  - This cluster-ordering contains info equiv to the density-based clusterings 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: Some Extension from DBSCAN

---

- Index-based:
  - $k = \text{number of dimensions}$
  - $N = 20$
  - $p = 75\%$
  - $M = N(1-p) = 5$
  - Complexity:  $O(N \log N)$
- Core Distance
- Reachability Distance

# Core Distance

---

**Definition 5:** (core-distance of an object p)

Let  $p$  be an object from a database  $D$ , let  $\varepsilon$  be a distance value, let  $N_\varepsilon(p)$  be the  $\varepsilon$ -neighborhood of  $p$ , let  $MinPts$  be a natural number and let  $MinPts\text{-distance}(p)$  be the distance from  $p$  to its  $MinPts$ ' neighbor. Then, the *core-distance* of  $p$  is defined as  $core-distance_{\varepsilon, MinPts}(p) =$

$$\begin{cases} \text{UNDEFINED, if } Card(N_\varepsilon(p)) < MinPts \\ MinPts\text{-distance}(p), \text{ otherwise} \end{cases}$$

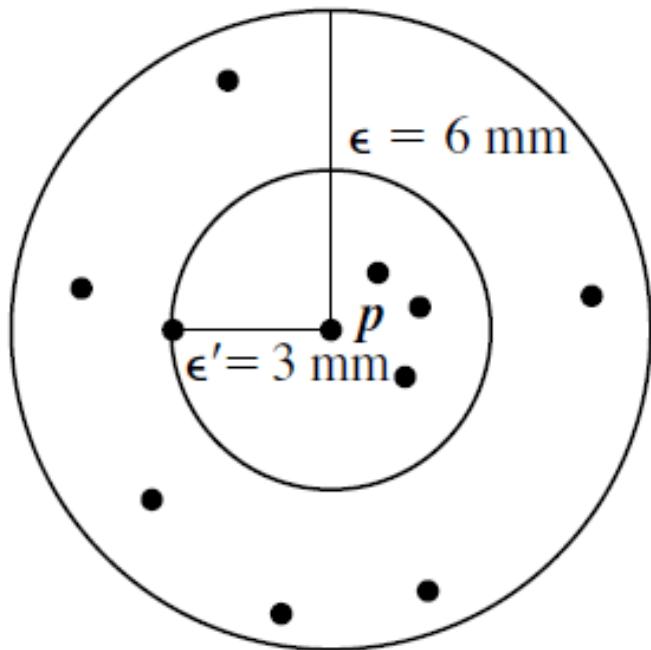
# Reachability Distance

---

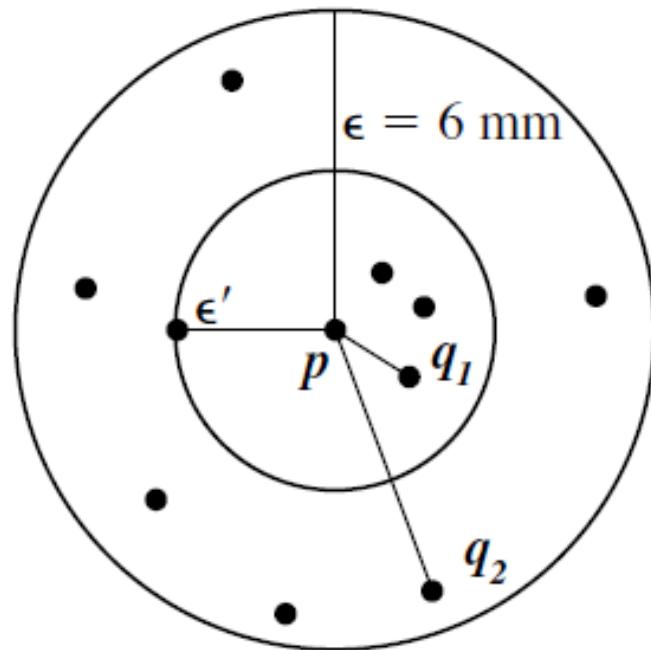
**Definition 6:** (reachability-distance object  $p$  w.r.t. object  $o$ )

Let  $p$  and  $o$  be objects from a database  $D$ , let  $N_\varepsilon(o)$  be the  $\varepsilon$ -neighborhood of  $o$ , and let  $MinPts$  be a natural number. Then, the *reachability-distance* of  $p$  with respect to  $o$  is defined as  $reachability-distance_{\varepsilon, MinPts}(p, o) =$

$$\begin{cases} \text{UNDEFINED, if } |N_\varepsilon(o)| < MinPts \\ \max(\text{core-distance}(o), \text{distance}(o, p)), \text{otherwise} \end{cases}$$



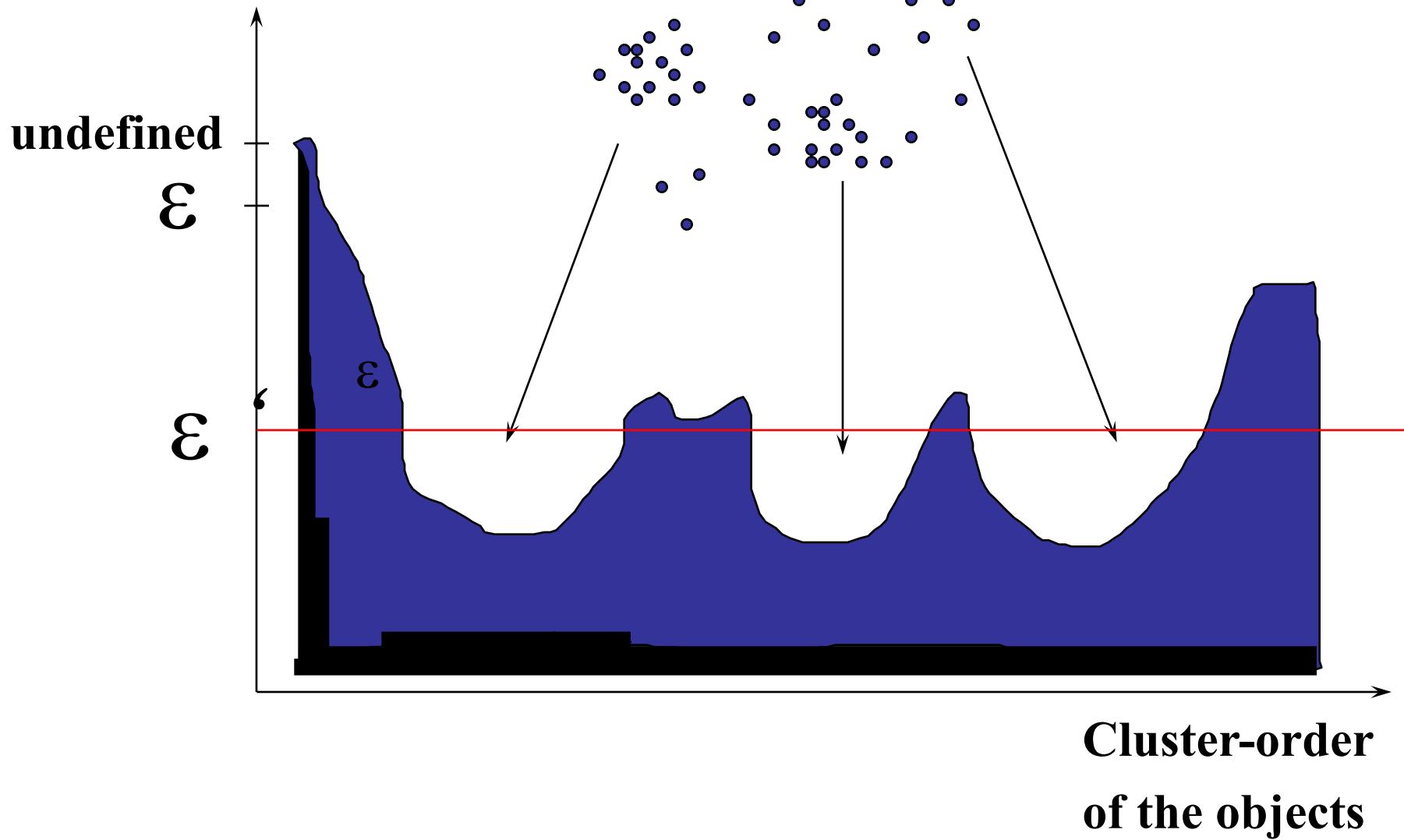
Core-distance of  $p$



Reachability-distance  $(p, q_1) = \epsilon' = 3 \text{ mm} \square$   
Reachability-distance  $(p, q_2) = d(p, q_2)$

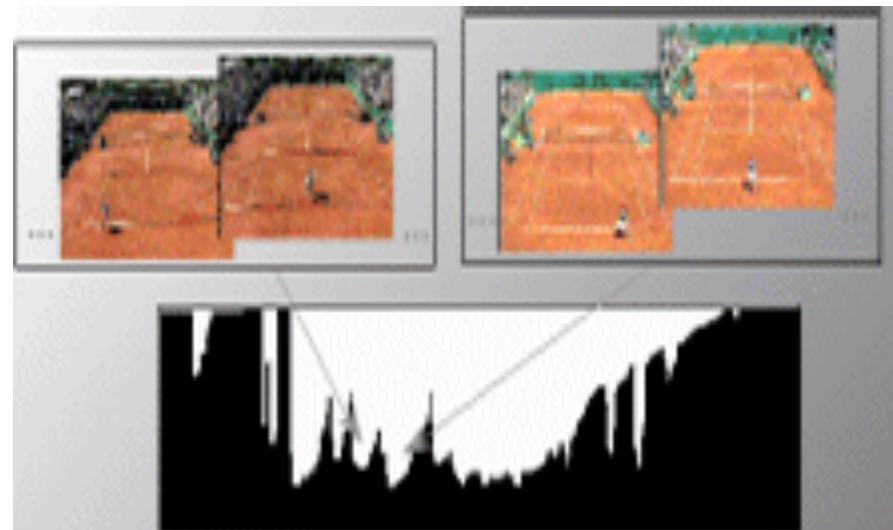
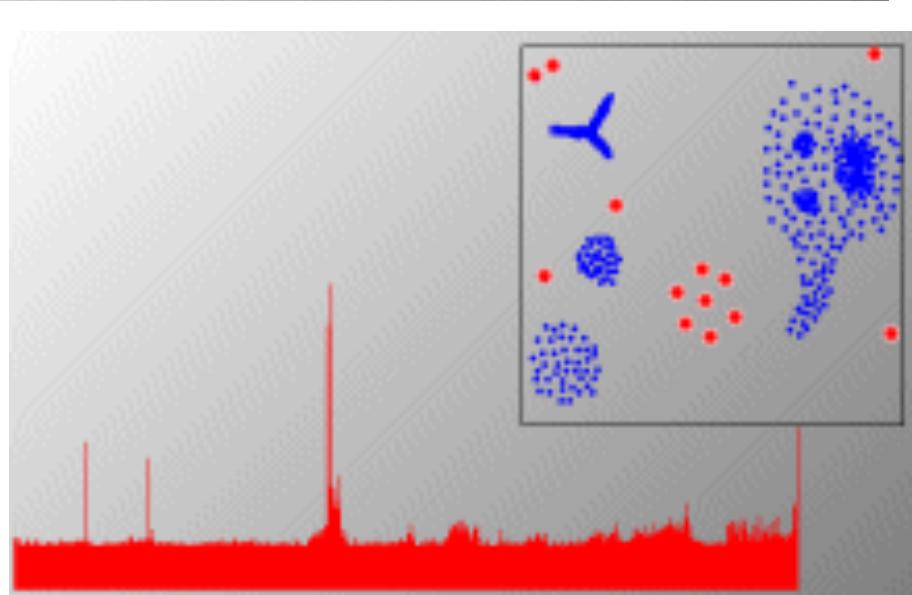
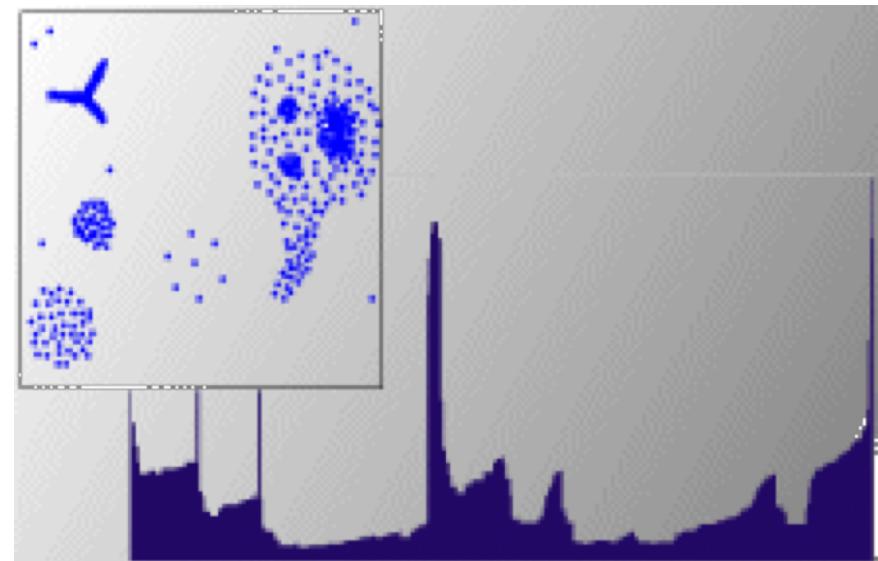
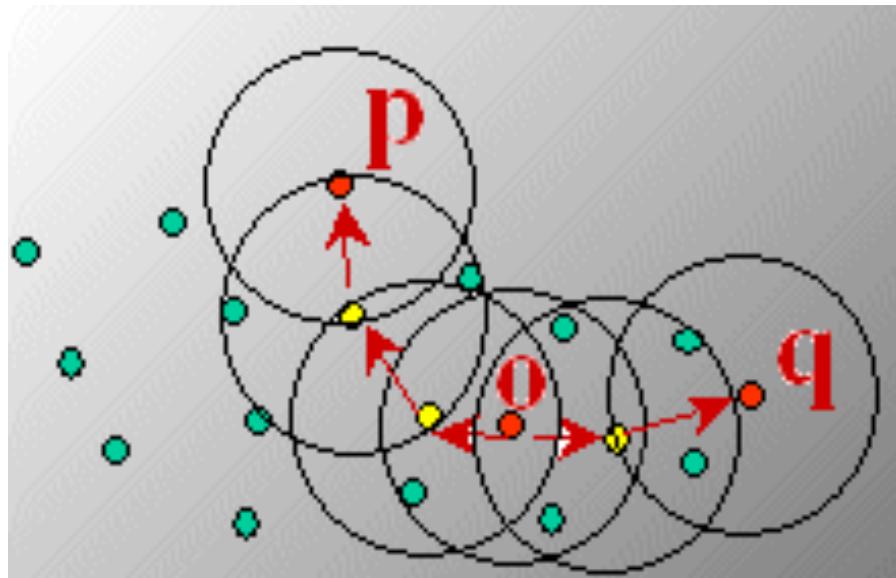
Figure 10.16: OPTICS terminology. Based on [ABKS99].

## Reachability -distance



Cluster-order  
of the objects

# Density-Based Clustering: OPTICS & Its Applications



# OPTICS Online Demo

---

- <http://www.dbs.informatik.uni-muenchen.de/Forschung/KDD/Clustering/OPTICS/Demo/>

# DENCLUE: Using Statistical Density Functions

- DENsity-based CLUstEring by Hinneburg & Keim (KDD'98)
- Using statistical density functions:

$$f_{Gaussian}(x, y) = e^{-\frac{d(x, y)^2}{2\sigma^2}}$$

influence of y  
on x

$$f_{Gaussian}^D(x) = \sum_{i=1}^N e^{-\frac{d(x, x_i)^2}{2\sigma^2}}$$

total influence  
on x

$$\nabla f_{Gaussian}^D(x, x_i) = \sum_{i=1}^N (x_i - x) \cdot e^{-\frac{d(x, x_i)^2}{2\sigma^2}}$$

gradient of x in  
the direction of  
 $x_i$

- 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., DBSCAN)
  - But needs a large number of parameters

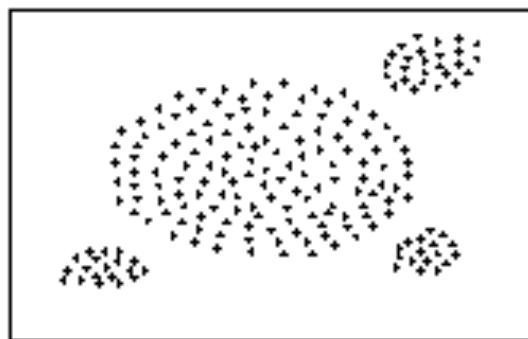
# Denclue: Technical Essence

---

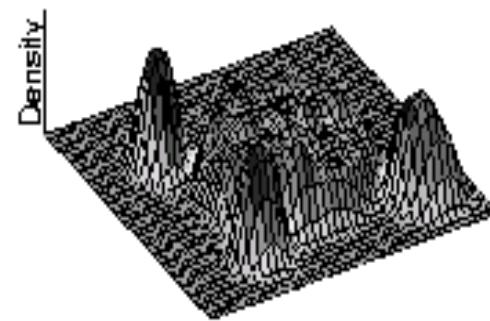
- Uses grid cells but only keeps information about grid cells that do actually contain data points and manages these cells in a tree-based access structure
- Influence function: describes the impact of a data point within its neighborhood
- Overall density of the data space can be calculated as the sum of the influence function of all data points
- Clusters can be determined mathematically by identifying density attractors
- Density attractors are local maximal of the overall density function
- Center defined clusters: assign to each density attractor the points density attracted to it
- Arbitrary shaped cluster: merge density attractors that are connected through paths of high density ( $>$  threshold)

# Density Attractor

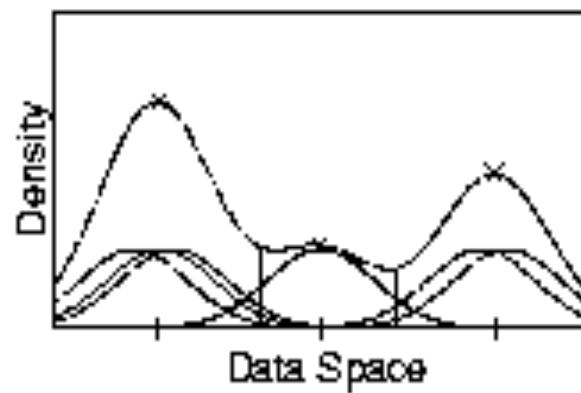
---



(a) Data Set



(c) Gaussian



# Center-Defined and Arbitrary

---

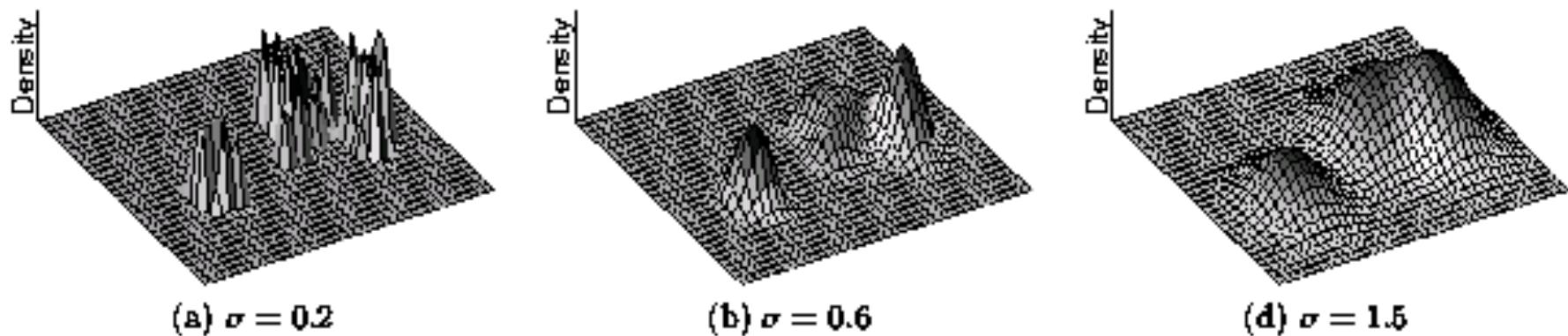


Figure 3: Example of Center-Defined Clusters for different  $\sigma$

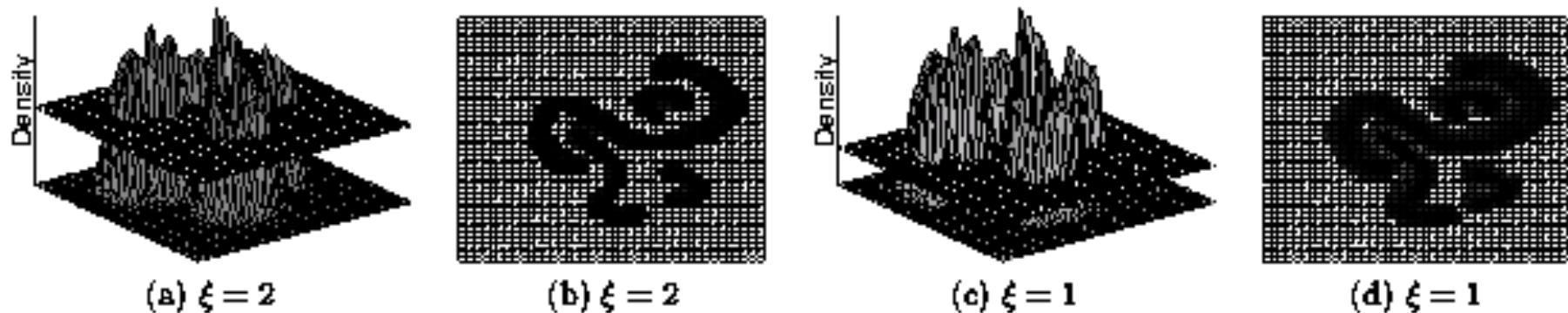


Figure 4: Example of Arbitrary-Shape Clusters for different  $\xi$

# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

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

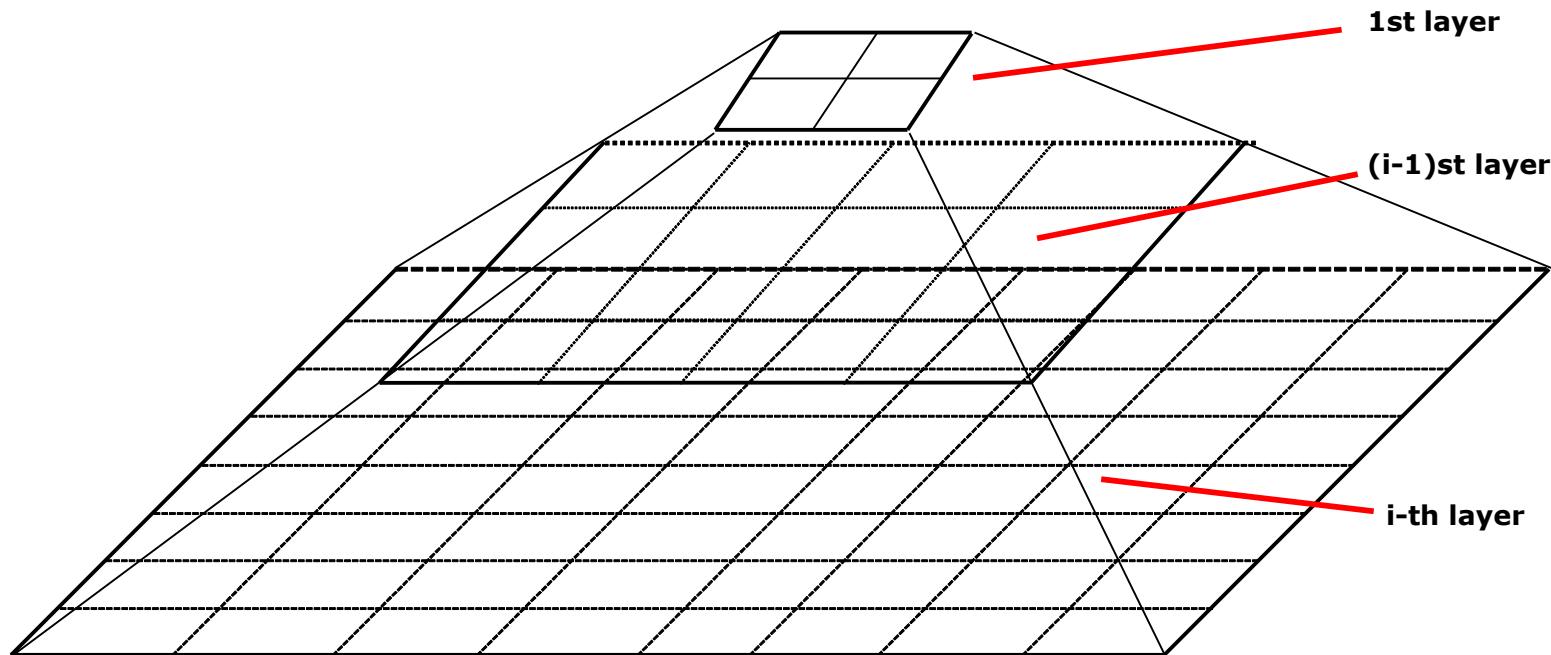
# Grid-Based Clustering Method

---

- Using multi-resolution grid data structure
- Several interesting methods
  - **STING** (a SStatistical INformation Grid approach) by Wang, Yang and Muntz (1997)
  - **WaveCluster** by Sheikholeslami, Chatterjee, and Zhang (VLDB'98)
    - A multi-resolution clustering approach using wavelet method
  - **CLIQUE**: Agrawal, et al. (SIGMOD'98)
    - Both grid-based and subspace clustering

# STING: A Statistical Information Grid Approach

- 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
  - *count, mean, s, min, max*
  - type of distribution—*normal, uniform*, etc.
- Use a top-down approach to answer spatial data queries
- Start from a pre-selected layer—typically with a small number of cells
- For each cell in the current level compute the confidence interval

# STING Algorithm and Its Analysis

---

- Remove the irrelevant cells from further consideration
- When finish examining the current layer, proceed to the next lower level
- Repeat this process until the bottom layer is reached
- Advantages:
  - Query-independent, easy to parallelize, incremental update
  - $O(K)$ , where  $K$  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

# CLIQUE (Clustering In QUEst)

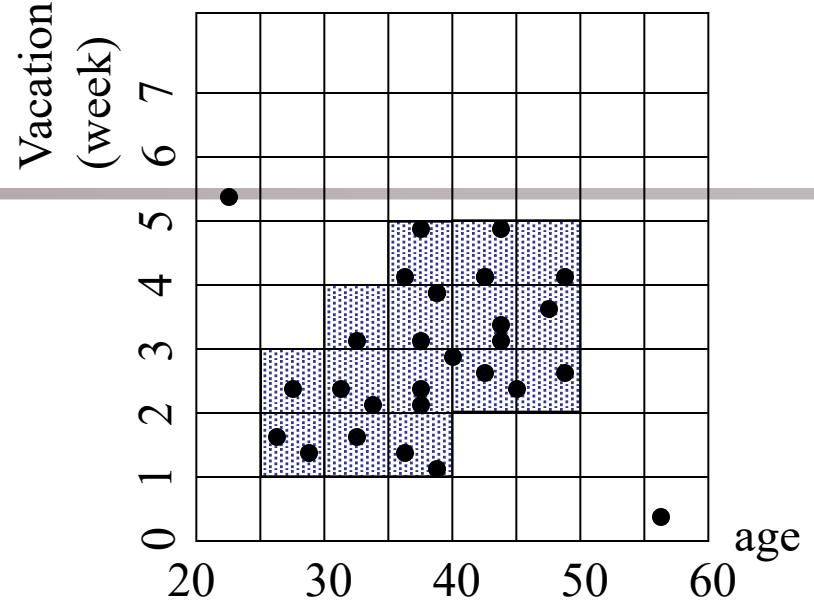
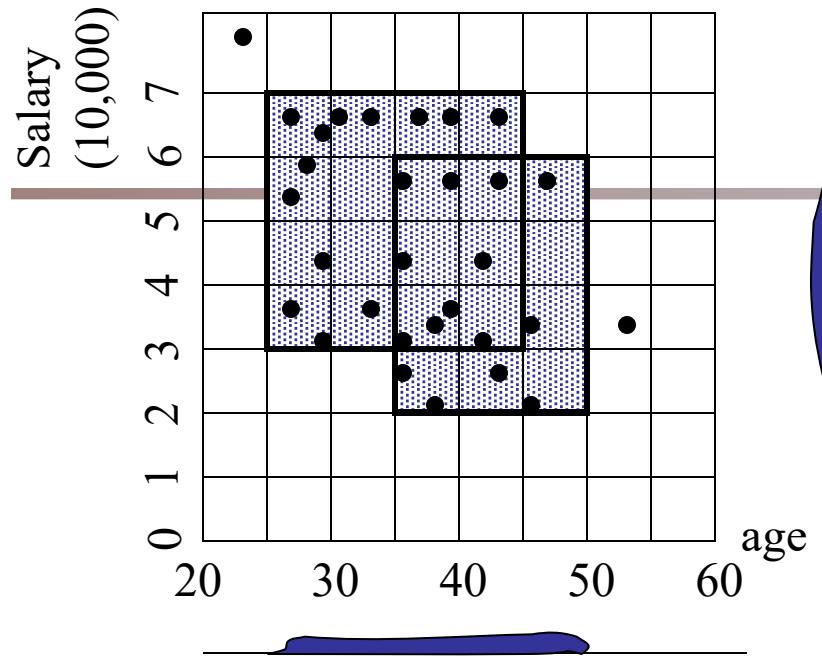
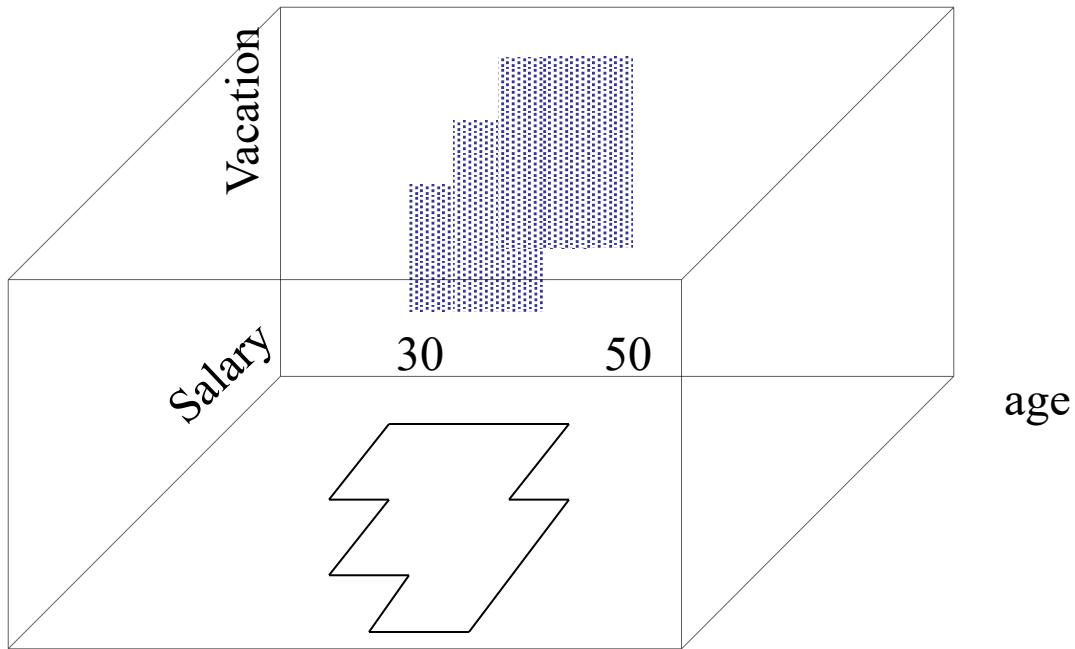
---

- 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 interval
  - 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
- 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


 $\tau = 3$ 


# **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
  - The accuracy of the clustering result may be degraded at the expense of simplicity of the method

# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

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

# Assessing Clustering Tendency

---

- Assess if non-random structure exists in the data by measuring the probability that the data is generated by a uniform data distribution
- Test spatial randomness by statistic test: Hopkins Static
  - Given a dataset D regarded as a sample of a random variable o, determine how far away o is from being uniformly distributed in the data space
  - Sample  $n$  points,  $p_1, \dots, p_n$ , uniformly from D. For each  $p_i$ , find its nearest neighbor in D:  $x_i = \min\{dist(p_i, v)\}$  where  $v$  in D
  - Sample  $n$  points,  $q_1, \dots, q_n$ , uniformly from D. For each  $q_i$ , find its nearest neighbor in  $D - \{q_i\}$ :  $y_i = \min\{dist(q_i, v)\}$  where  $v$  in D and  $v \neq q_i$
  - Calculate the Hopkins Statistic:
    - If D is uniformly distributed,  $\sum x_i$  and  $\sum y_i$  will be close to each other and H is close to 0.5. If D is clustered, H is close to 1

# Determine the Number of Clusters

---

- Empirical method
  - # of clusters  $\approx \sqrt{n}/2$  for a dataset of  $n$  points
- Elbow method
  - Use the turning point in the curve of sum of within cluster variance w.r.t the # 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:  $Q(C, C_g)$ , for a clustering  $C$  given the ground truth  $C_g$ .
- $Q$  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

# Chapter 10. 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





# CS512-Spring 2011: An Introduction

---

- Coverage
  - Cluster Analysis: Chapter 11
  - Outlier Detection: Chapter 12
  - Mining Sequence Data: BK2: Chapter 8
  - Mining Graphs Data: BK2: Chapter 9
  - Social and Information Network Analysis
    - BK2: Chapter 9
    - Partial coverage: Mark Newman: “Networks: An Introduction”, Oxford U., 2010
    - Scattered coverage: Easley and Kleinberg, “Networks, Crowds, and Markets: Reasoning About a Highly Connected World”, Cambridge U., 2010
    - Recent research papers
  - Mining Data Streams: BK2: Chapter 8
- Requirements
  - One research project
  - One class presentation (15 minutes)
  - Two homeworks (no programming assignment)
  - Two midterm exams (no final exam)

# References (1)

---

- R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. SIGMOD'98
- M. R. Anderberg. Cluster Analysis for Applications. Academic Press, 1973.
- M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure, SIGMOD'99.
- Beil F., Ester M., Xu X.: "Frequent Term-Based Text Clustering", KDD'02
- M. M. Breunig, H.-P. Kriegel, R. Ng, J. Sander. LOF: Identifying Density-Based Local Outliers. SIGMOD 2000.
- M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases. KDD'96.
- M. Ester, H.-P. Kriegel, and X. Xu. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. SSD'95.
- D. Fisher. Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2:139-172, 1987.
- D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamic systems. VLDB'98.
- V. Ganti, J. Gehrke, R. Ramakrishnan. CACTUS Clustering Categorical Data Using Summaries. KDD'99.

## References (2)

---

- D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamic systems. In Proc. VLDB'98.
- S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. SIGMOD'98.
- S. Guha, R. Rastogi, and K. Shim. ROCK: A robust clustering algorithm for categorical attributes. In ICDE'99, pp. 512-521, Sydney, Australia, March 1999.
- A. Hinneburg, D.I A. Keim: An Efficient Approach to Clustering in Large Multimedia Databases with Noise. KDD'98.
- A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Prentice Hall, 1988.
- G. Karypis, E.-H. Han, and V. Kumar. CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. COMPUTER, 32(8): 68-75, 1999.
- L. Kaufman and P. J. Rousseeuw. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.
- E. Knorr and R. Ng. Algorithms for mining distance-based outliers in large datasets. VLDB'98.

## References (3)

---

- G. J. McLachlan and K.E. Bkasford. Mixture Models: Inference and Applications to Clustering. John Wiley and Sons, 1988.
- R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. VLDB'94.
- L. Parsons, E. Haque and H. Liu, Subspace Clustering for High Dimensional Data: A Review, SIGKDD Explorations, 6(1), June 2004
- E. Schikuta. Grid clustering: An efficient hierarchical clustering method for very large data sets. Proc. 1996 Int. Conf. on Pattern Recognition.,
- G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. VLDB'98.
- A. K. H. Tung, J. Han, L. V. S. Lakshmanan, and R. T. Ng. Constraint-Based Clustering in Large Databases, ICDT'01.
- A. K. H. Tung, J. Hou, and J. Han. Spatial Clustering in the Presence of Obstacles, ICDE'01
- H. Wang, W. Wang, J. Yang, and P.S. Yu. Clustering by pattern similarity in large data sets, SIGMOD' 02.
- W. Wang, Yang, R. Muntz, STING: A Statistical Information grid Approach to Spatial Data Mining, VLDB'97.
- T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH : An efficient data clustering method for very large databases. SIGMOD'96.
- Xiaoxin Yin, Jiawei Han, and Philip Yu, "[LinkClus: Efficient Clustering via Heterogeneous Semantic Links](#)", in Proc. 2006 Int. Conf. on Very Large Data Bases (VLDB'06), Seoul, Korea, Sept. 2006.

# Chapter 10. Cluster Analysis: Basic Concepts and Methods

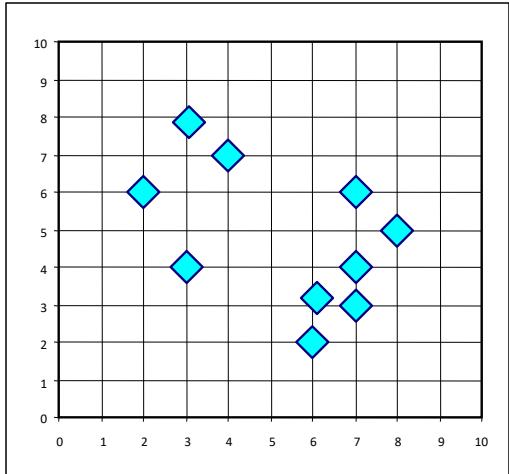
---

- Cluster Analysis: Basic Concepts
  - What Is Cluster Analysis?
  - What is Good Clustering? Measuring the Quality of Clustering
  - Major categories of clustering methods
- Clustering structures
  - Calculating Distance between Clusters
- Partitioning Methods
  - $k$ -Means: A Classical Partitioning Method
  - Alternative Methods:  $k$ -Medoids,  $k$ -Median, and its Variations
- Hierarchical Methods
  - Agglomerative and Divisive Hierarchical Clustering
  - BIRCH: A Hierarchical, Micro-Clustering Approach
  - Chameleon: A Hierarchical Clustering Algorithm Using Dynamic Modeling
- Density-Based Methods
  - DBSCAN and OPTICS: Density-Based Clustering Based on Connected Regions
  - DENCLUE: Clustering Based on Density Distribution Functions
- Link-Based Cluster Analysis
  - SimRank: Exploring Links in Cluster Analysis
  - LinkClus: Scalability in Link-Based Cluster Analysis
- Grid-Based Methods
  - STING: SStatistical INformation Grid
  - WaveCluster: Clustering Using Wavelet Transformation
  - CLIQUE: A Dimension-Growth Subspace Clustering Method
- Summary

---

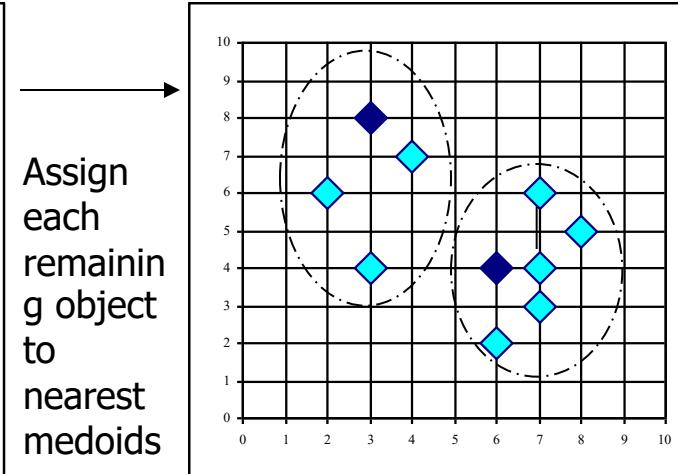
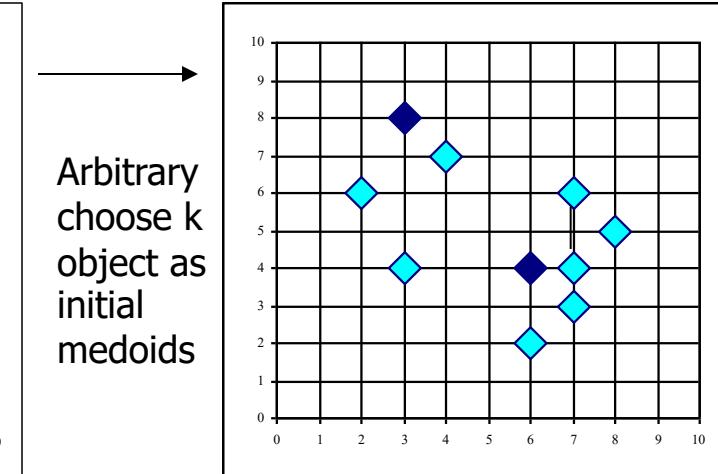
**Slides unused in class**

# A Typical K-Medoids Algorithm (PAM)



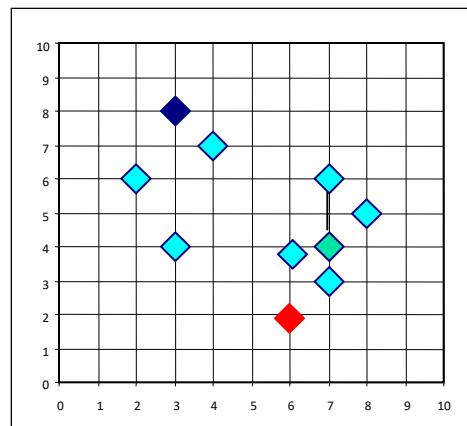
K=2

**Do loop**  
**Until no change**

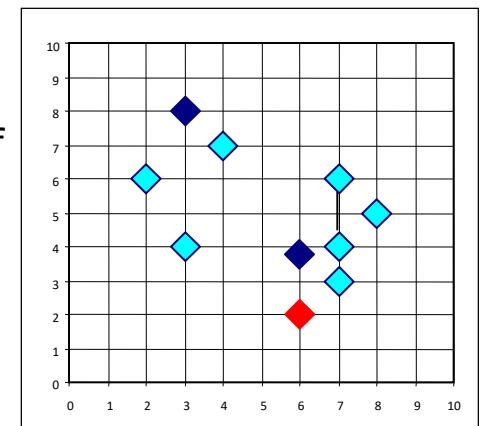


Total Cost = 26

Randomly select a nonmedoid object,  $O_{random}$



Compute total cost of swapping



Swapping  $O$  and  $O_{random}$   
If quality is improved.

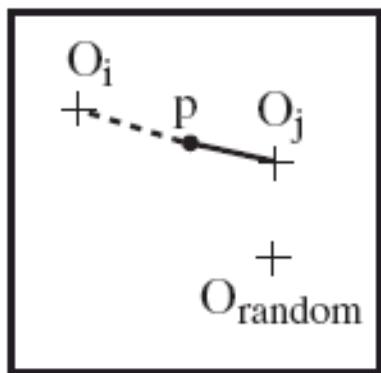
# PAM (Partitioning Around Medoids) (1987)

---

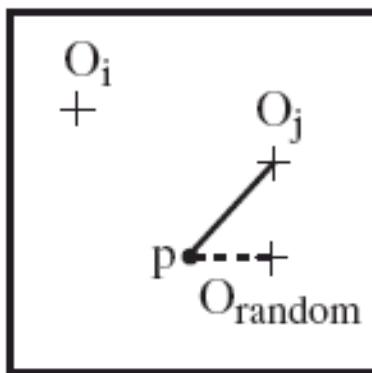
- PAM (Kaufman and Rousseeuw, 1987), built in Splus
- Use real object to represent the cluster
  - Select  $k$  representative objects arbitrarily
  - For each pair of non-selected object  $h$  and selected object  $i$ , calculate the total swapping cost  $TC_{ih}$
  - For each pair of  $i$  and  $h$ ,
    - If  $TC_{ih} < 0$ ,  $i$  is replaced by  $h$
    - Then assign each non-selected object to the most similar representative object
  - repeat steps 2-3 until there is no change

# PAM Clustering: Finding the Best Cluster Center

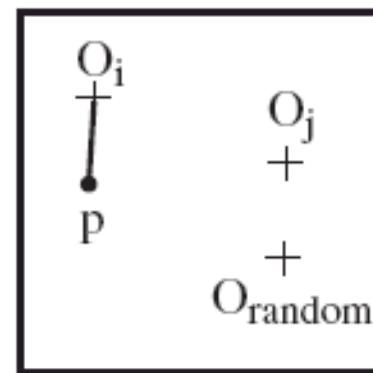
- Case 1:  $p$  currently belongs to  $o_j$ . If  $o_j$  is replaced by  $o_{\text{random}}$  as a representative object and  $p$  is the closest to one of the other representative object  $o_i$ , then  $p$  is reassigned to  $o_i$



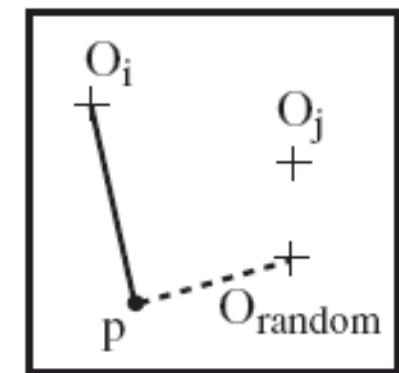
1. Reassigned to  $O_i$



2. Reassigned to  $O_{\text{random}}$



3. No change



4. Reassigned to  $O_{\text{random}}$

- data object
- + cluster center
- before swapping
- after swapping

# 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

→ Sampling-based method

CLARA(Clustering LARge Applications)

# **CLARA (Clustering Large Applications) (1990)**

---

- CLARA (Kaufmann and Rousseeuw in 1990)
  - Built in statistical analysis packages, such as SPlus
  - It draws *multiple samples* of the data set, applies *PAM* on each sample, and gives the best clustering as the output
- Strength: deals with larger data sets than *PAM*
- 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) (1994)**

---

- CLARANS (A Clustering Algorithm based on Randomized Search) (Ng and Han'94)
  - Draws sample of neighbors dynamically
  - The clustering process can be presented as searching a graph where every node is a potential solution, that is, a set of  $k$  medoids
  - If the local optimum is found, *it* starts with new randomly selected node in search for a new local optimum
- Advantages: More efficient and scalable than both *PAM* and *CLARA*
- Further improvement: Focusing techniques and spatial access structures (Ester et al.'95)

# **ROCK: Clustering Categorical Data**

---

- ROCK: RObust Clustering using linKs
  - S. Guha, R. Rastogi & K. Shim, ICDE'99
- Major ideas
  - Use links to measure similarity/proximity
  - Not distance-based
- Algorithm: sampling-based clustering
  - Draw random sample
  - Cluster with links
  - Label data in disk
- Experiments
  - Congressional voting, mushroom data

# Similarity Measure in ROCK

---

- Traditional measures for categorical data may not work well, e.g., Jaccard coefficient
- Example: Two groups (clusters) of transactions
  - C<sub>1</sub>. <a, b, c, d, e>: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}
  - C<sub>2</sub>. <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}
- Jaccard co-efficient may lead to wrong clustering result
  - C<sub>1</sub>: 0.2 ({a, b, c}, {b, d, e}) to 0.5 ({a, b, c}, {a, b, d})
  - C<sub>1</sub> & C<sub>2</sub>: could be as high as 0.5 ({a, b, c}, {a, b, f})
- Jaccard co-efficient-based similarity function:
  - Ex. Let T<sub>1</sub> = {a, b, c}, T<sub>2</sub> = {c, d, e}

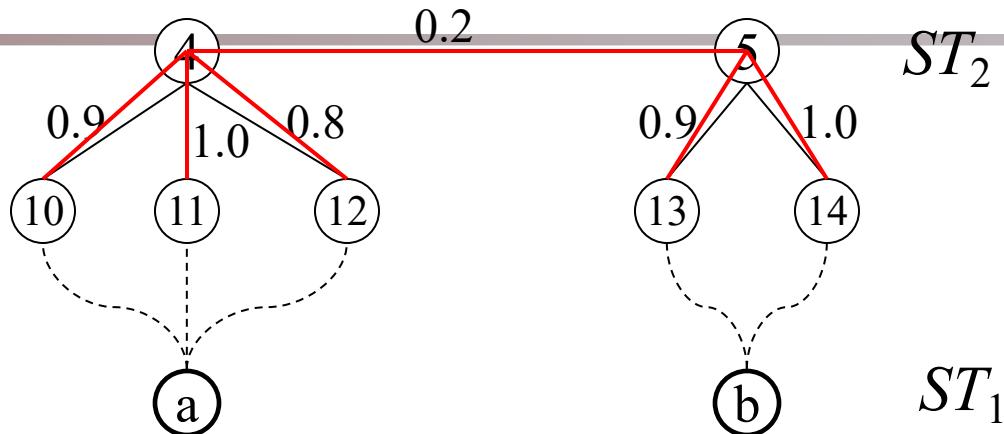
$$Sim(T_1, T_2) = \frac{|T_1 \cap T_2|}{|T_1 \cup T_2|}$$

$$Sim(T_1, T_2) = \frac{|\{c\}|}{|\{a, b, c, d, e\}|} = \frac{1}{5} = 0.2$$

# Link Measure in ROCK

- Clusters
  - $C_1: \langle a, b, c, d, e \rangle: \{a, b, c\}, \{a, b, d\}, \{a, b, e\}, \{a, c, d\}, \{a, c, e\}, \{a, d, e\}, \{b, c, d\}, \{b, c, e\}, \{b, d, e\}, \{c, d, e\}$
  - $C_2: \langle a, b, f, g \rangle: \{a, b, f\}, \{a, b, g\}, \{a, f, g\}, \{b, f, g\}$
- Neighbors
  - Two transactions are neighbors if  $\text{sim}(T_1, T_2) > \text{threshold}$
  - Let  $T_1 = \{a, b, c\}$ ,  $T_2 = \{c, d, e\}$ ,  $T_3 = \{a, b, f\}$ 
    - $T_1$  connected to:  $\{a, b, d\}, \{a, b, e\}, \{a, c, d\}, \{a, c, e\}, \{b, c, d\}, \{b, c, e\}, \{a, b, f\}, \{a, b, g\}$
    - $T_2$  connected to:  $\{a, c, d\}, \{a, c, e\}, \{a, d, e\}, \{b, c, e\}, \{b, d, e\}, \{b, c, d\}$
    - $T_3$  connected to:  $\{a, b, c\}, \{a, b, d\}, \{a, b, e\}, \{a, b, g\}, \{a, f, g\}, \{b, f, g\}$
- Link Similarity
  - Link similarity between two transactions is the # of common neighbors
  - $\text{link}(T_1, T_2) = 4$ , since they have 4 common neighbors
    - $\{a, c, d\}, \{a, c, e\}, \{b, c, d\}, \{b, c, e\}$
  - $\text{link}(T_1, T_3) = 3$ , since they have 3 common neighbors
    - $\{a, b, d\}, \{a, b, e\}, \{a, b, g\}$

# Aggregation-Based Similarity Computation



For each node  $n_k \in \{n_{10}, n_{11}, n_{12}\}$  and  $n_l \in \{n_{13}, n_{14}\}$ , their path-based similarity  $sim_p(n_k, n_l) = s(n_k, n_4) \cdot s(n_4, n_5) \cdot s(n_5, n_l)$ .

$$sim(n_a, n_b) = \frac{\sum_{k=10}^{12} s(n_k, n_4)}{3} \cdot s(n_4, n_5) \cdot \frac{\sum_{l=13}^{14} s(n_l, n_5)}{2} = 0.171$$

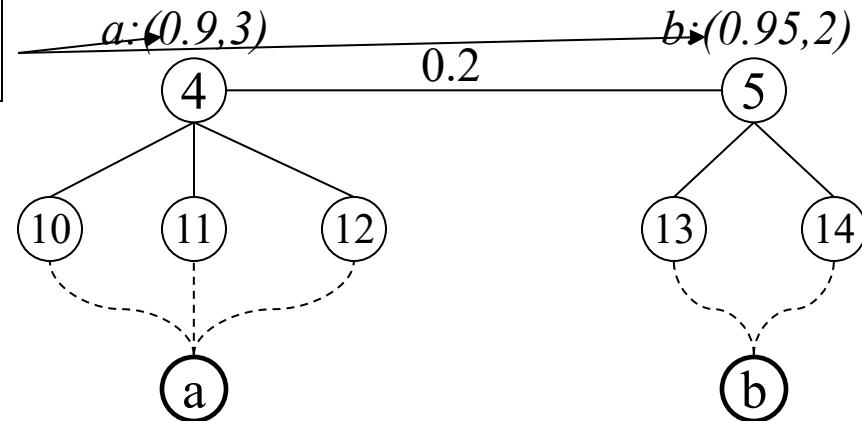
takes  $O(3+2)$  time

After aggregation, we reduce quadratic time computation to linear time computation.

# Computing Similarity with Aggregation

Average similarity  
and total weight

$sim(n_a, n_b)$  can be computed  
from aggregated similarities



$$\begin{aligned} sim(n_a, n_b) &= \text{avg\_sim}(n_a, n_4) \times s(n_4, n_5) \times \text{avg\_sim}(n_b, n_5) \\ &= 0.9 \times 0.2 \times 0.95 = 0.171 \end{aligned}$$

To compute  $sim(n_a, n_b)$ :

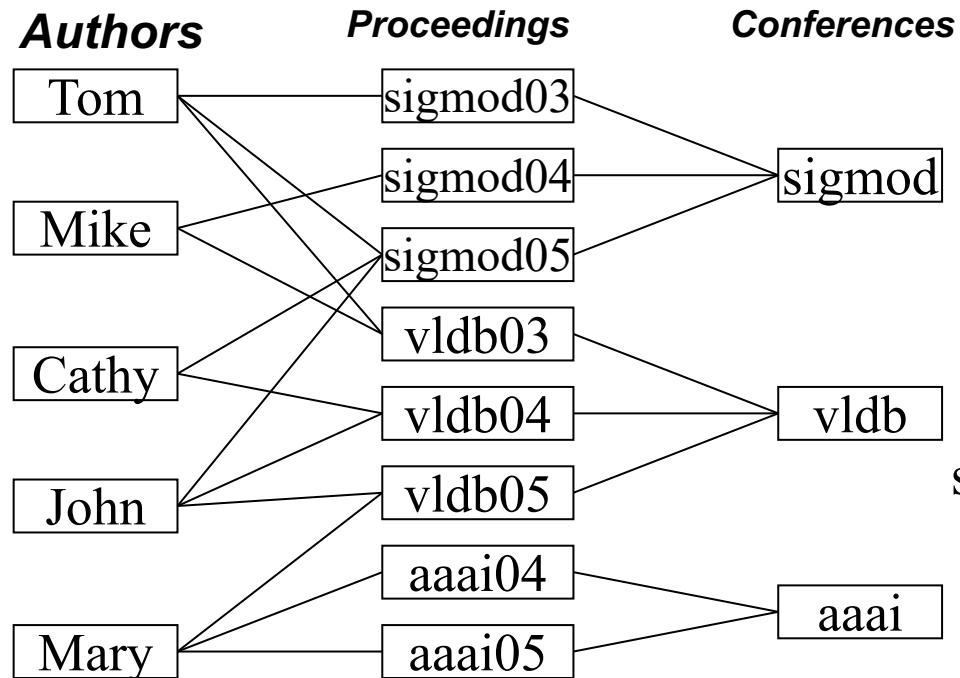
- Find all pairs of sibling nodes  $n_i$  and  $n_j$ , so that  $n_a$  linked with  $n_i$  and  $n_b$  with  $n_j$ .
- Calculate similarity (and weight) between  $n_a$  and  $n_b$  w.r.t.  $n_i$  and  $n_j$ .
- Calculate weighted average similarity between  $n_a$  and  $n_b$  w.r.t. all such pairs.

# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

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

# Link-Based Clustering: Calculate Similarities Based On Links



Jeh & Widom, KDD'2002: *SimRank*

Two objects are similar if they are linked with the same or similar objects

- The similarity between two objects  $x$  and  $y$  is defined as the average similarity between objects linked with  $x$  and those with  $y$ :

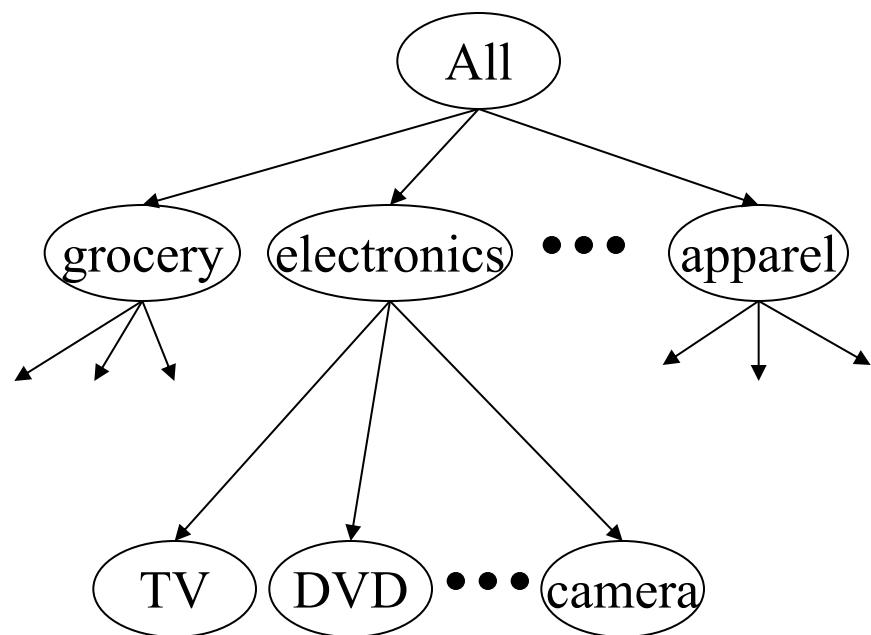
$$\text{sim}(a, b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} \text{sim}(I_i(a), I_j(b))$$

- Issue: Expensive to compute:
  - For a dataset of  $N$  objects and  $M$  links, it takes  $O(N^2)$  space and  $O(M^2)$  time to compute all similarities.

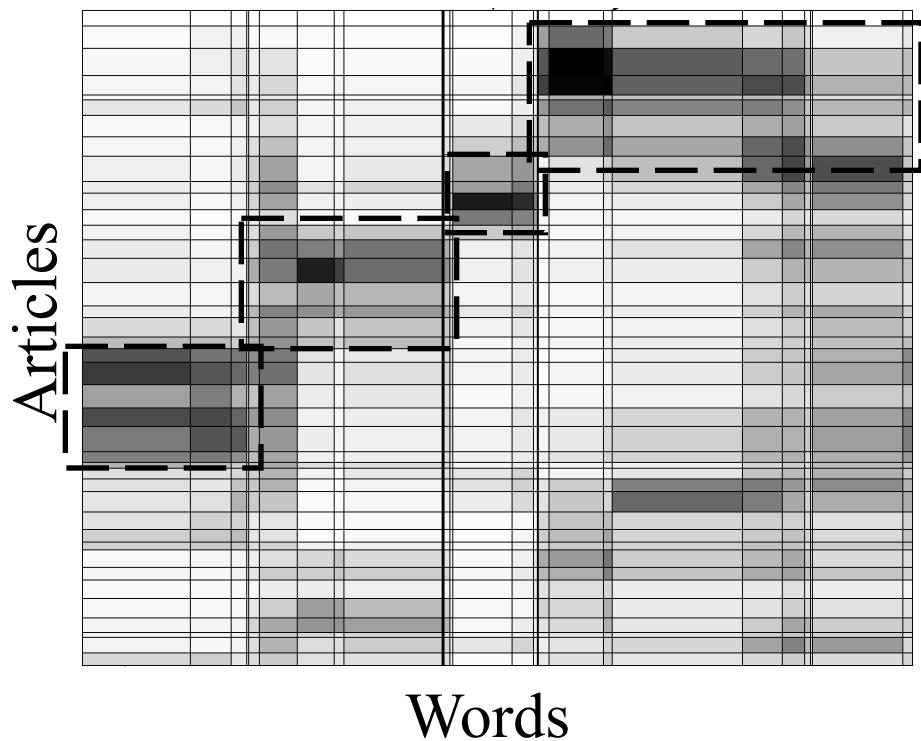
# Observation 1: Hierarchical Structures

- Hierarchical structures often exist naturally among objects (e.g., taxonomy of animals)

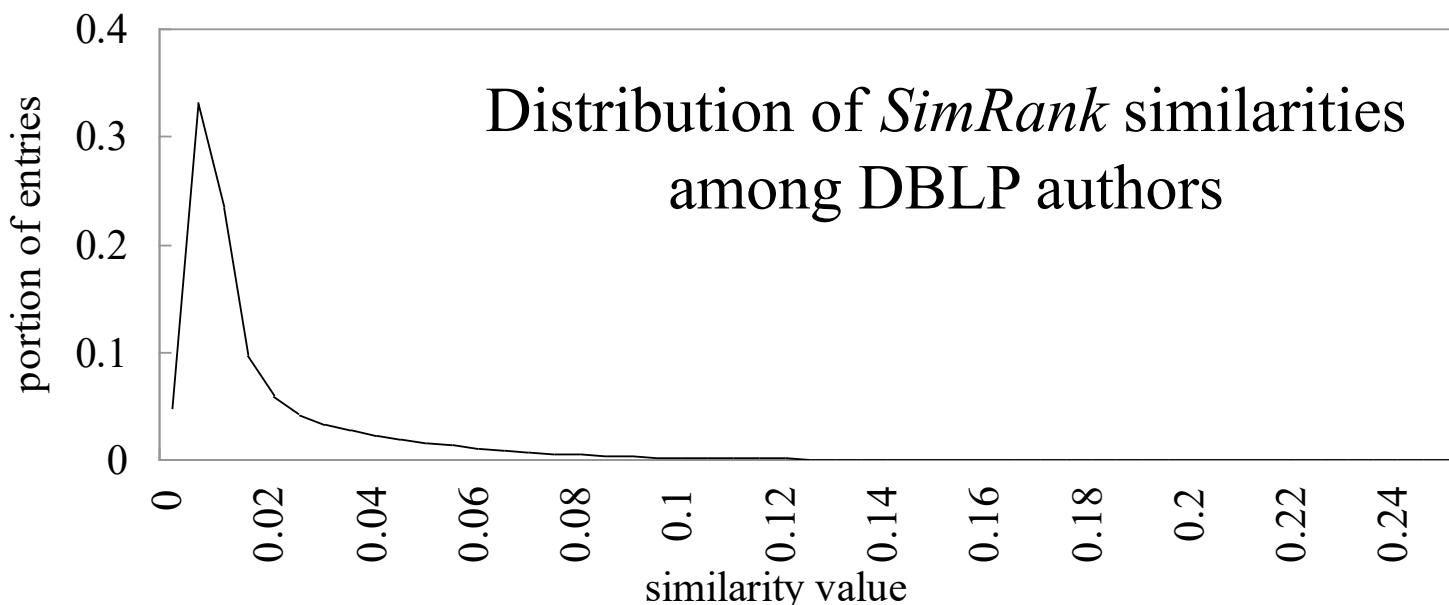
A hierarchical structure of products in Walmart



Relationships between articles and words (Chakrabarti, Papadimitriou, Modha, Faloutsos, 2004)

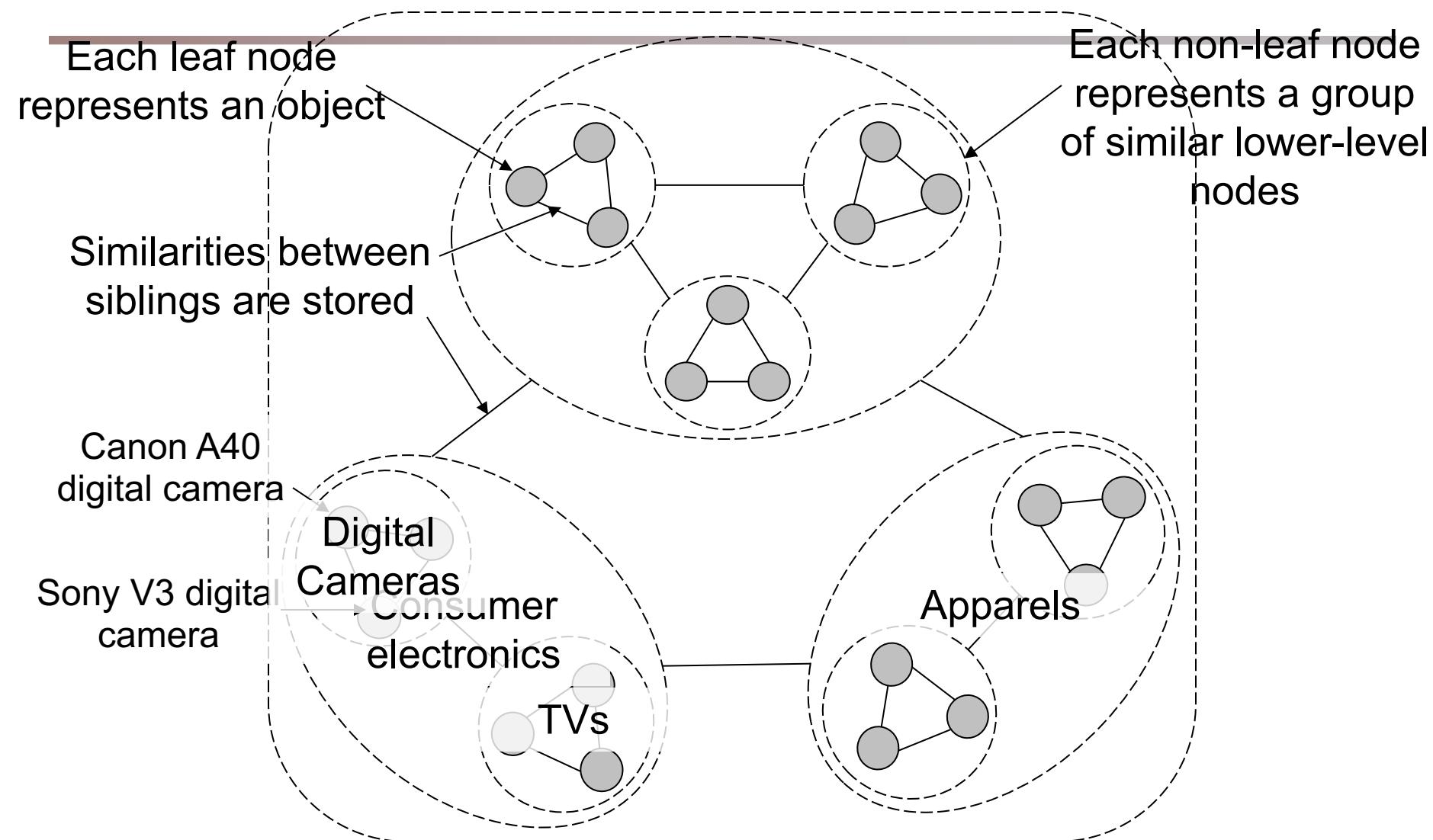


## Observation 2: Distribution of Similarity



- Power law distribution exists in similarities
  - 56% of similarity entries are in [0.005, 0.015]
  - 1.4% of similarity entries are larger than 0.1
  - Can we design a data structure that stores the significant similarities and compresses insignificant ones?

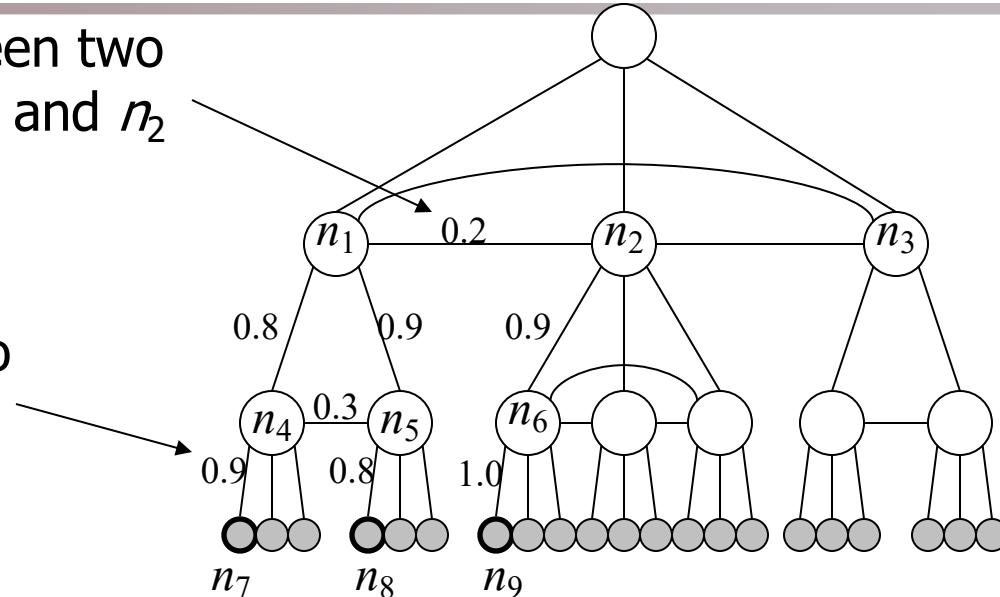
# A Novel Data Structure: SimTree



# Similarity Defined by SimTree

Similarity between two sibling nodes  $n_1$  and  $n_2$

Adjustment ratio for node  $n_7$



- Path-based node similarity
  - $sim_p(n_7, n_8) = s(n_7, n_4) \times s(n_4, n_5) \times s(n_5, n_8)$
- Similarity between two nodes is the average similarity between objects linked with them in other SimTrees
- Adjust/ ratio for  $x = \frac{\text{Average similarity between } x \text{ and all other nodes}}{\text{Average similarity between } x \text{'s parent and all other nodes}}$

# LinkClus: Efficient Clustering via Heterogeneous Semantic Links

---

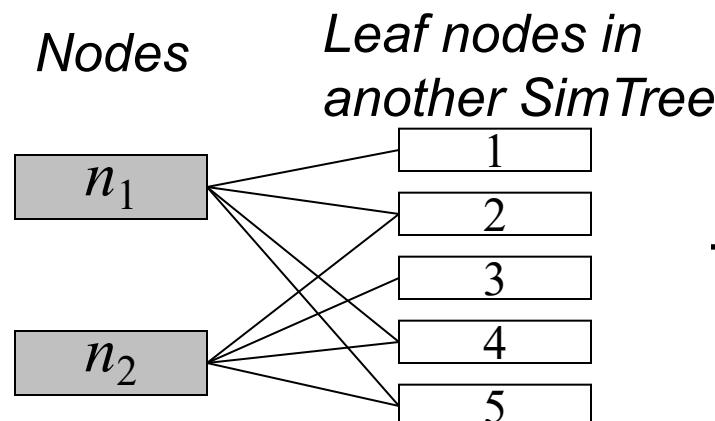
## Method

- Initialize a SimTree for objects of each type
- Repeat until stable
  - For each SimTree, update the similarities between its nodes using similarities in other SimTrees
    - Similarity between two nodes  $x$  and  $y$  is the average similarity between objects linked with them
  - Adjust the structure of each SimTree
    - Assign each node to the parent node that it is most similar to

For details: X. Yin, J. Han, and P. S. Yu, “LinkClus: Efficient Clustering via Heterogeneous Semantic Links”, VLDB'06

# Initialization of SimTrees

- Initializing a SimTree
  - Repeatedly find groups of tightly related nodes, which are merged into a higher-level node
- Tightness of a group of nodes
  - For a group of nodes  $\{n_1, \dots, n_k\}$ , its tightness is defined as the number of leaf nodes in other SimTrees that are connected to all of  $\{n_1, \dots, n_k\}$



The tightness of  $\{n_1, n_2\}$  is 3

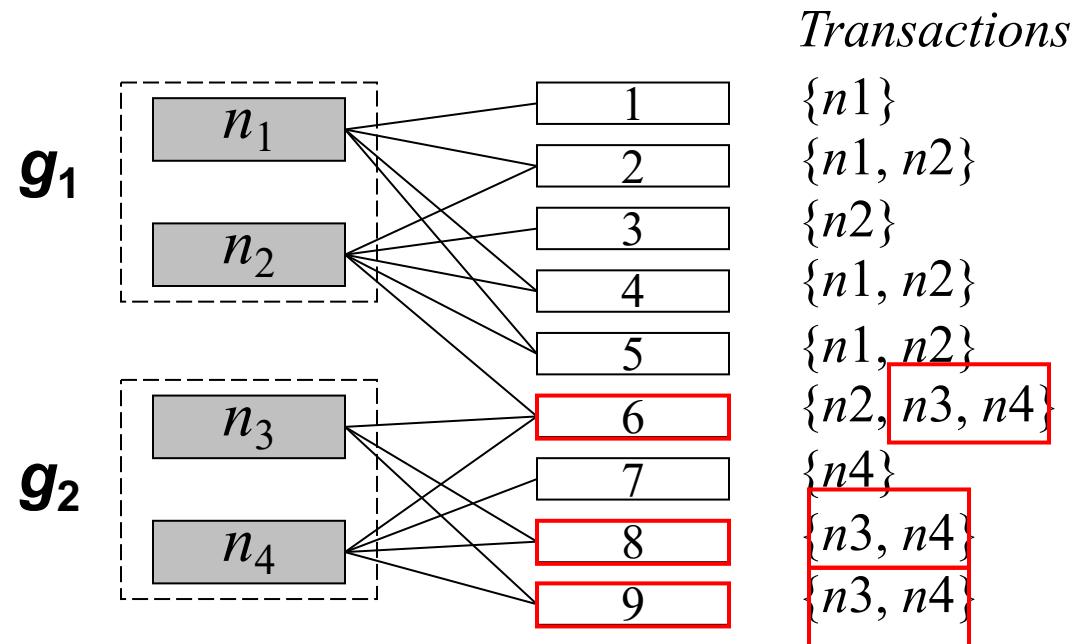
# Finding Tight Groups by Freq. Pattern Mining

- Finding tight groups

Reduced to

Frequent pattern mining

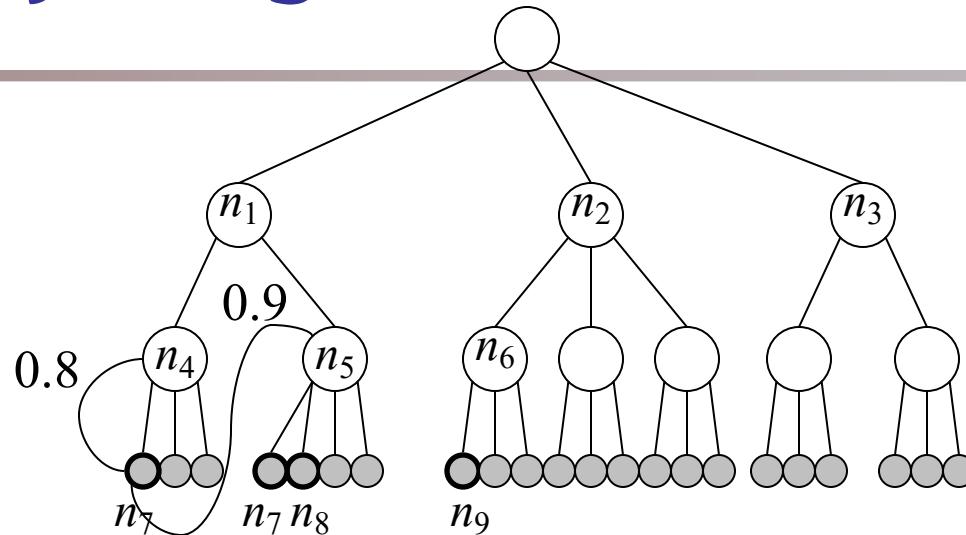
The tightness of a group of nodes is the support of a frequent pattern



- Procedure of initializing a tree
  - Start from leaf nodes (level-0)
  - At each level  $l$ , find non-overlapping groups of similar nodes with frequent pattern mining

# Adjusting SimTree Structures

---



- After similarity changes, the tree structure also needs to be changed
  - If a node is more similar to its parent's sibling, then move it to be a child of that sibling
  - Try to move each node to its parent's sibling that it is most similar to, under the constraint that each parent node can have at most  $c$  children

# Complexity

---

For two types of objects,  $N$  in each, and  $M$  linkages between them.

	Time	Space
Updating similarities	$O(M(\log N)^2)$	$O(M+N)$
Adjusting tree structures	$O(N)$	$O(N)$
<i>LinkClus</i>	$O(M(\log N)^2)$	$O(M+N)$
<i>SimRank</i>	$O(M^2)$	$O(N^2)$

# Experiment: Email Dataset

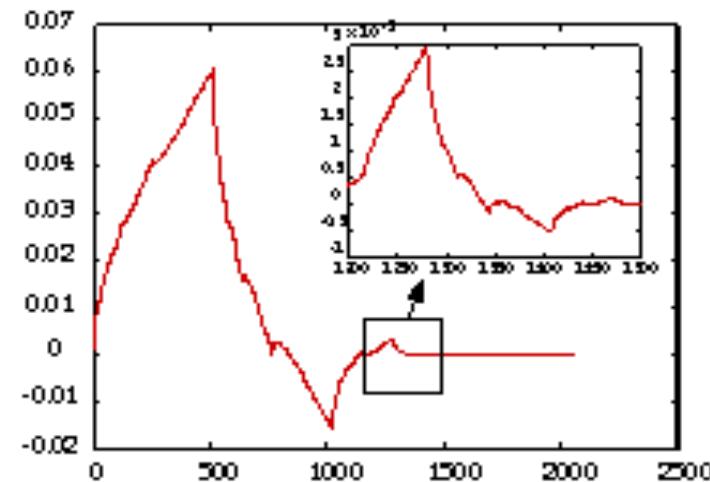
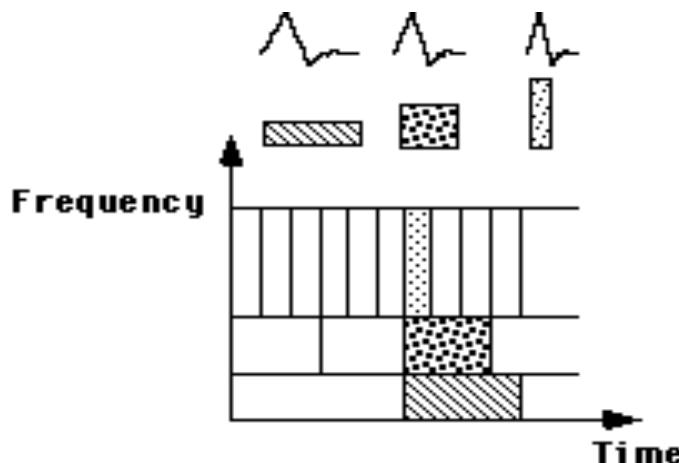
- F. Nielsen. Email dataset.  
[www.imm.dtu.dk/~rem/data/Email-1431.zip](http://www.imm.dtu.dk/~rem/data/Email-1431.zip)
- 370 emails on conferences, 272 on jobs, and 789 spam emails
- Accuracy: measured by manually labeled data
- Accuracy of clustering: % of pairs of objects in the same cluster that share common label

<i>Approach</i>	<i>Accuracy</i>	<i>time (s)</i>
LinkClus	0.8026	1579.6
SimRank	0.7965	39160
ReCom	0.5711	74.6
F-SimRank	0.3688	479.7
CLARANS	0.4768	8.55

- Approaches compared:
  - SimRank (Jeh & Widom, KDD 2002): Computing pair-wise similarities
  - SimRank with FingerPrints (F-SimRank): Fogaras & R'acz, WWW 2005
    - pre-computes a large sample of random paths from each object and uses samples of two objects to estimate SimRank similarity
  - ReCom (Wang et al. SIGIR 2003)
    - Iteratively clustering objects using cluster labels of linked objects

# WaveCluster: Clustering by Wavelet Analysis (1998)

- Sheikholeslami, Chatterjee, and Zhang (VLDB'98)
- A multi-resolution clustering approach which applies wavelet transform to the feature space; both grid-based and density-based
- Wavelet transform: A signal processing technique that decomposes a signal into different frequency sub-band
  - Data are transformed to preserve relative distance between objects at different levels of resolution
  - Allows natural clusters to become more distinguishable



# The WaveCluster Algorithm

---

- How to apply wavelet transform to find clusters
  - Summarizes the data by imposing a multidimensional grid structure onto data space
  - These multidimensional spatial data objects are represented in a n-dimensional feature space
  - Apply wavelet transform on feature space to find the dense regions in the feature space
  - Apply wavelet transform multiple times which result in clusters at different scales from fine to coarse
- Major features:
  - Complexity  $O(N)$
  - Detect arbitrary shaped clusters at different scales
  - Not sensitive to noise, not sensitive to input order
  - Only applicable to low dimensional data

# Quantization & Transformation

- Quantize data into m-D grid structure  
then wavelet transform
  - a) scale 1: high resolution
  - b) scale 2: medium resolution
  - c) scale 3: low resolution

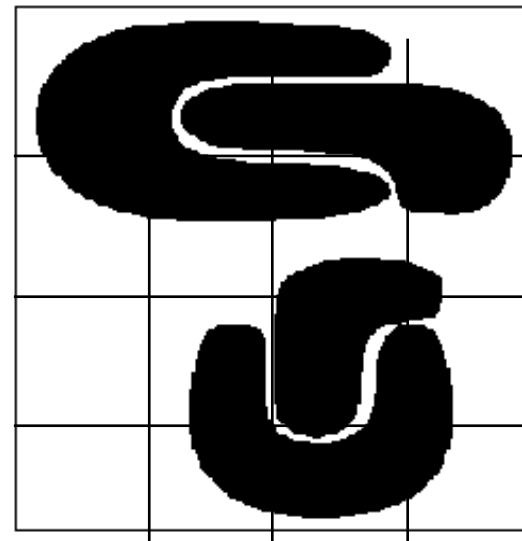
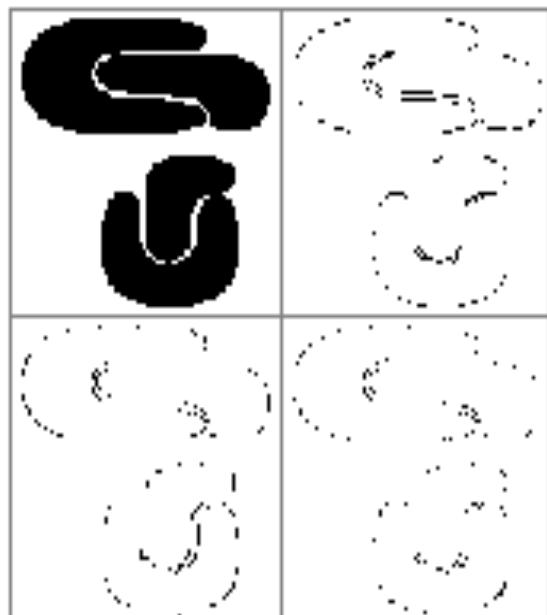


Figure 1: A sample 2-dimensional feature space.



a)



b)



c)