Data Mining:

Concepts and Techniques

— Chapter 7 —

Jiawei Han

Department of Computer Science

University of Illinois at Urbana-Champaign

www.cs.uiuc.edu/~hanj

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods

What is Cluster Analysis?

- Cluster: a collection of data objects
  - *Similar* to one another within the same cluster
  - *Dissimilar* to the objects in other clusters

  → distance (or similarity) measures
- Cluster analysis
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Examples of Clustering Applications

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- 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

Requirements of Clustering in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- High dimensionality
- Incorporation of user-specified constraints
- Interpretability and usability

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods

Data Structures

- Data matrix
  - *n* objects, *p* attributes
  - (two modes)
  - One row represents one object

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

- Dissimilarity matrix
  - Distance table
  - (one mode)

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ : & : & : & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

Type of data in clustering analysis

- Interval-scaled variables
  - Continuous measurements (weight, temperature, …)
- Binary variables
  - Variables with 2 states (on/off, yes/no)
- Nominal variables
  - A generalization of the binary variable in that it can take more than 2 states (color/red,yellow,blue,green)
- Ordinal
  - ranking is important (e.g. medals(gold,silver,bronze))
- Ratio variables
  - a positive measurement on a nonlinear scale (growth)
- Variables of mixed types

Interval-valued variables

- Sometimes we need to standardize the data
  - Calculate the mean absolute deviation:

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + ... + |x_{nf} - m_f|)$$

where $m_f = \frac{1}{n}(x_{1f} + x_{2f} + ... + x_{nf})$.

  - Calculate the standardized measurement (*z-score*)

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

Distances between objects

Distances are normally used to measure the similarity or dissimilarity between two data objects

- Properties
  - $d(i,j) \geq 0$
  - $d(i,i) = 0$
  - $d(i,j) = d(j,i)$
  - $d(i,j) \leq d(i,k) + d(k,j)$

Distances between objects

- Euclidean distance:

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+...+|x_{i_p}-x_{j_p}|^2)}$$

- Manhattan distance:

$$d(i,j)=|x_{i_1}-x_{j_1}|+|x_{i_2}-x_{j_2}|+...+|x_{i_p}-x_{j_p}|$$

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two *p*-dimensional data objects,

Distances between objects

- *Minkowski distance*:

$$d(i,j) = \sqrt[q]{(|x_{i_1} - x_{j_1}|^q + |x_{i_2} - x_{j_2}|^q + \ldots + |x_{i_p} - x_{j_p}|^q)}$$

q is a positive integer

- If *q = 1, d* is Manhattan distance
- If *q = 2, d* is Euclidean distance

Binary Variables

- symmetric binary variables: both states are equally important; 0/1
- asymmetric binary variables: one state is more important than the other (e.g. outcome of disease test); 1 is the important state, 0 the other

Contingency tables for Binary Variables

		Object *j*		
		1	0	*sum*
Object ***i***	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

a: number of attributes having 1 for object i and 1 for object j
b: number of attributes having 1 for object i and 0 for object j
c: number of attributes having 0 for object i and 1 for object j
d: number of attributes having 0 for object i and 0 for object j
p = a+b+c+d

Distance measure for symmetric binary variables

		Object j		
		1	0	sum
Object i	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

Distance measure for asymmetric binary variables

		Object j		
		1	0	sum
Object i	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

$$d(i,j) = \frac{b+c}{a+b+c}$$

Jaccard coefficient = 1- d(i,j) = $sim_{Jaccard}(i,j) = \frac{a}{a+b+c}$

Dissimilarity between Binary Variables

- Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

  - gender is a symmetric attribute
  - the remaining attributes are asymmetric binary

    → distance based on these
  - let the values Y and P be set to 1, and the value N be set to 0

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Nominal or Categorical Variables

- Method 1: Simple matching
  - *m*: # of matches, *p*: total # of variables

$$d(i, j) = \frac{p - m}{p}$$

- Method 2: use a large number of binary variables
  - creating a new asymmetric binary variable for each of the *M* nominal states

Ordinal Variables

- An ordinal variable can be discrete or continuous
- Order is important, e.g., rank
- Can be treated like interval-scaled
  - replace x_{if} by their rank $r_{if} \in \{1, \ldots, M_f\}$
  - map the range of each variable onto [0, 1] by replacing *i*-th object in the *f*-th variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

  - compute the dissimilarity using methods for interval-scaled variables

Ratio-Scaled Variables

- Ratio-scaled variable: a positive measurement on a nonlinear scale, approximately at exponential scale,
  such as Ae^{Bt} or Ae^{-Bt}
- Methods:
  - treat them like interval-scaled variables—*not a good choice!* (why?—the scale can be distorted)
  - apply logarithmic transformation

    $$y_{if} = log(x_{if})$$

  - treat them as continuous ordinal data, treat their rank as interval-scaled

Variables of Mixed Types

- A database may contain all the six types of variables
  - symmetric binary, asymmetric binary, nominal, ordinal, interval and ratio
- One may use a weighted formula to combine their effects:

$$d(i, j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}}$$

  - *f* is binary or nominal:

    $d_{ij}^{(f)} = 0$ if $x_{if} = x_{jf}$, or $d_{ij}^{(f)} = 1$ otherwise
  - *f* is interval-based: use the (normalized) distance
  - *f* is ordinal or ratio-scaled
    - compute ranks r_{if} and
    - and treat z_{if} as interval-scaled

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

  - delta(i,j) = 0 iff (i) x-value is missing or (ii) x-values are 0 and f asymmetric binary attribute

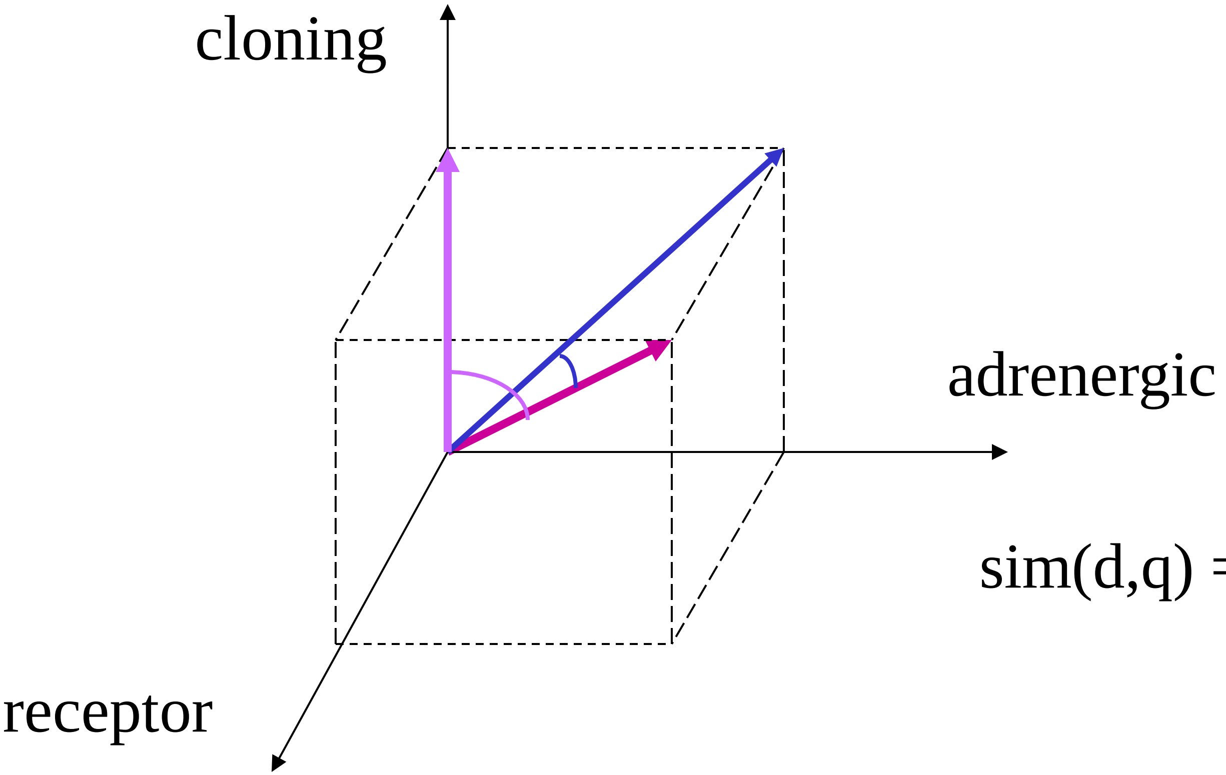
Vector Objects

- Vector objects: keywords in documents, gene features in micro-arrays, etc.
- Broad applications: information retrieval, biologic taxonomy, etc.
- Cosine measure

$$s(\vec{X}, \vec{Y}) = \frac{\vec{X}^t \cdot \vec{Y}}{|\vec{X}||\vec{Y}|},$$

$\vec{X}^t$ is a transposition of vector $\vec{X}$, $|\vec{X}|$ is the Euclidean normal of vector $\vec{X}$,

Vector model for information retrieval (simplified)



Doc1 (1,1,0)

Doc2 (0,1,0)

Q (1,1,1)

$$\mathrm{sim}(d,q) = \frac{d \cdot q}{|d| \times |q|}$$

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods

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, ROCK, CHAMELEON
- Density-based approach:
  - Based on connectivity and density functions
  - Typical methods: DBSCAN, OPTICS, DenClue

Major Clustering Approaches (II)

- Grid-based approach:
  - based on a multiple-level granularity structure
  - Typical methods: STING, WaveCluster, CLIQUE
- 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: pCluster
- User-guided or constraint-based:
  - Clustering by considering user-specified or application-specific constraints
  - Typical methods: COD (obstacles), constrained clustering

Typical Alternatives to Calculate the Distance between Clusters

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

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

- Centroid: the "middle" of a cluster

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

- Radius: square root of average distance from any point of the cluster to its centroid

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

- Diameter: square root of average mean squared distance between all pairs of points in the cluster

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

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods

Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database ***D*** of ***n*** objects into a set of ***k*** clusters, s.t., min sum of squared distance

$$\Sigma_{m=1}^{k}\Sigma_{t_{mi}\in Km}(C_m - t_{mi})^2$$

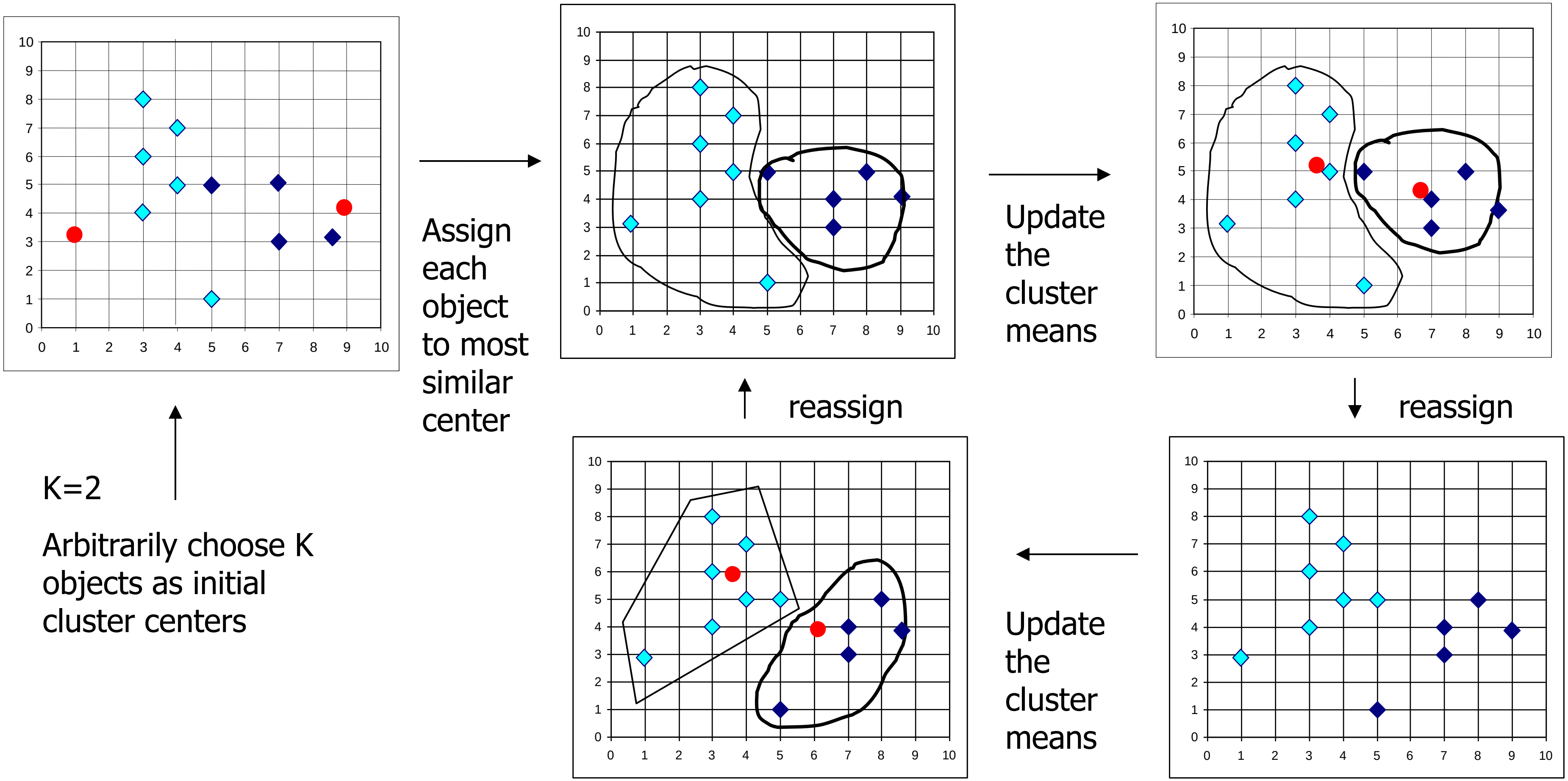
- Given a *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - *k-means* (MacQueen'67): Each cluster is represented by the center of the cluster
  - *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*, data D
- 1. arbitrarily choose *k* objects as initial cluster centers
- 2. Repeat
  - Assign each object to the cluster to which the object is most similar based on mean values of the objects in the cluster
  - Update cluster means (calculate mean value of the objects for each cluster)
- Until no change

The *K-Means* Clustering Method

- Example



K=2

Arbitrarily choose K objects as initial cluster centers

Assign each object to most similar center

Update the cluster means

reassign

reassign

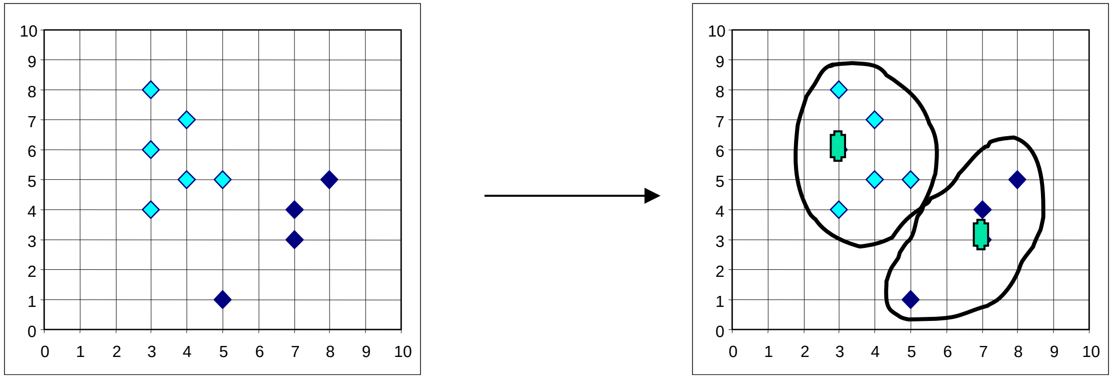
Update the cluster means

Comments on the *K-Means* Method

- Strength: *Relatively 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))
- Comment: Often terminates at a *local optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*
- Weakness
    - Applicable only when *mean* is defined, then what about categorical data?
    - Need to specify *k,* the *number* of clusters, in advance
    - Unable to handle noisy data and *outliers*
    - Not suitable to discover clusters with *non-convex shapes*

What Is the Problem of the K-Means Method?

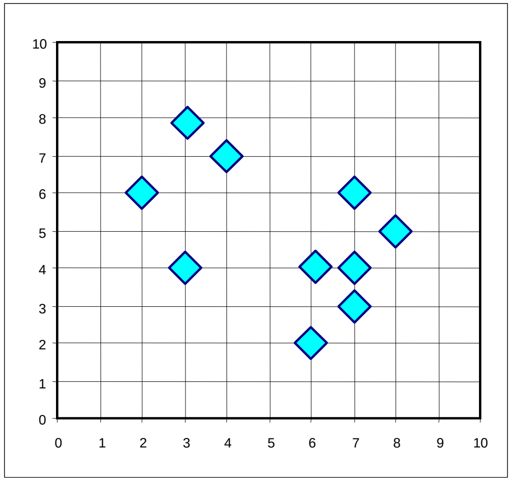
- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data.
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most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



The *K-Medoids* Clustering Method

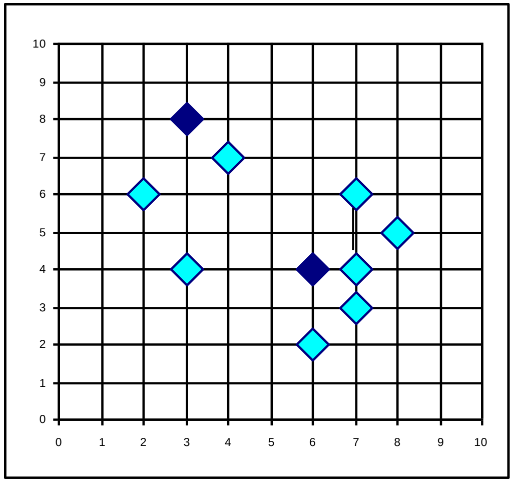
- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - *PAM* works effectively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

A Typical K-Medoids Algorithm (PAM) – basic idea



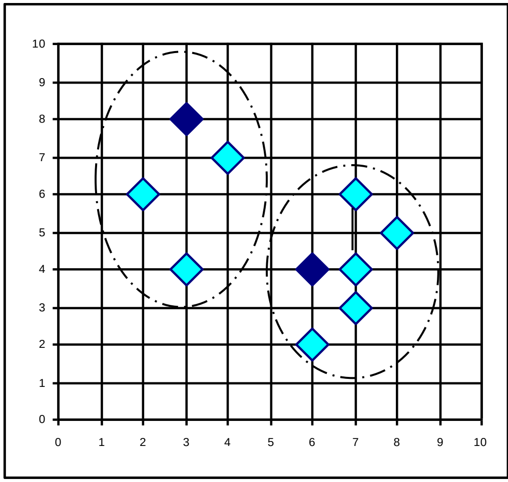
K=2

Arbitrary choose k objects as initial medoids



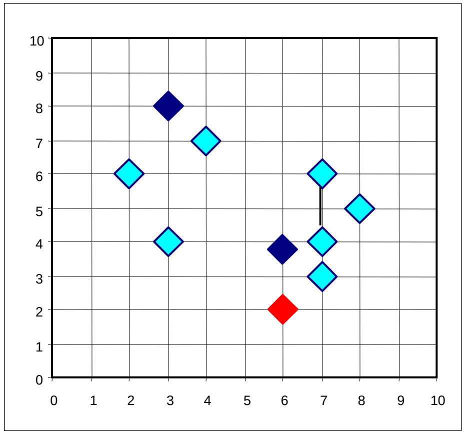
Assign each remaining object to nearest medoids

Total Cost = 20



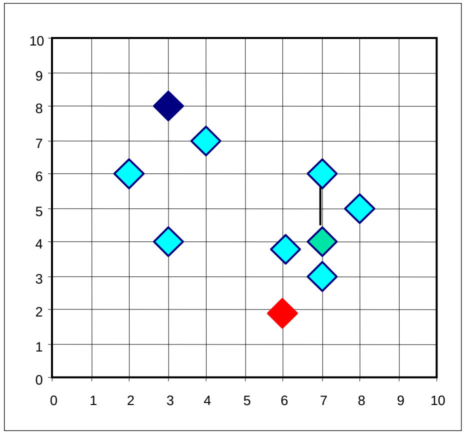
All pairs

Randomly select a nonmedoid object,O_{ramdom}



Compute total cost of swapping

Total Cost = 26



Swapping O and O_{ramdom}

For **best pair**

If quality is improved.

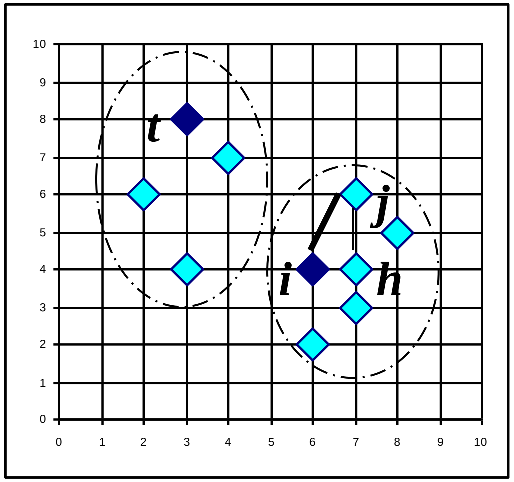
Do loop

Until no change

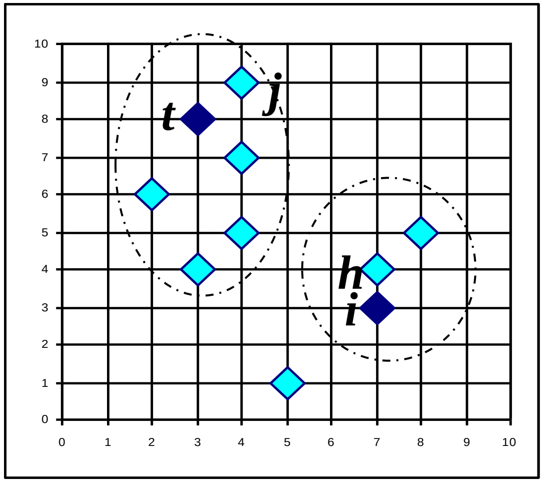
PAM (Partitioning Around Medoids) (1987)

- PAM (Kaufman and Rousseeuw, 1987)
- Algorithm
  - Select ***k*** representative objects arbitrarily
  - For each pair of non-selected object ***h*** and selected object ***i***, calculate the total swapping cost $\boldsymbol{TC_{ih}}$
  - Select a pair ***i*** and ***h***, which corresponds to the minimum swapping cost
    - 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

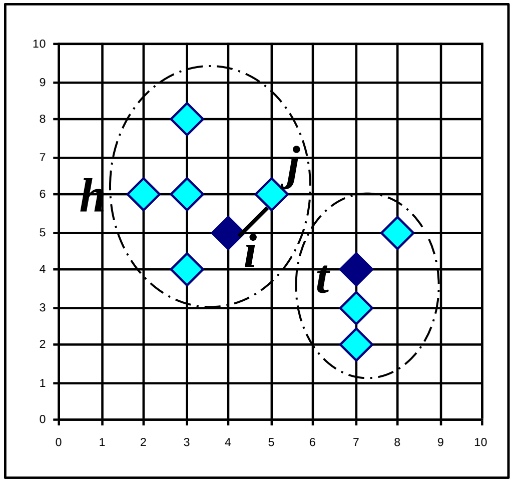
PAM Clustering: Total swapping cost $TC_{ih}=\sum_j C_{jih}$



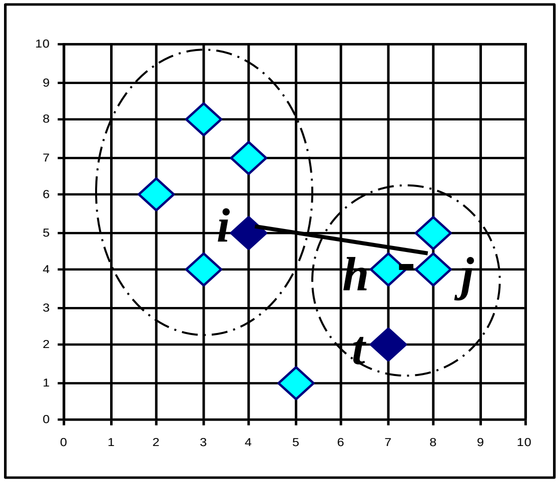
$C_{jih} = d(j, h) - d(j, i)$



$C_{jih} = 0$



$C_{jih} = d(j, t) - d(j, i)$



$C_{jih} = d(j, h) - d(j, t)$

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)

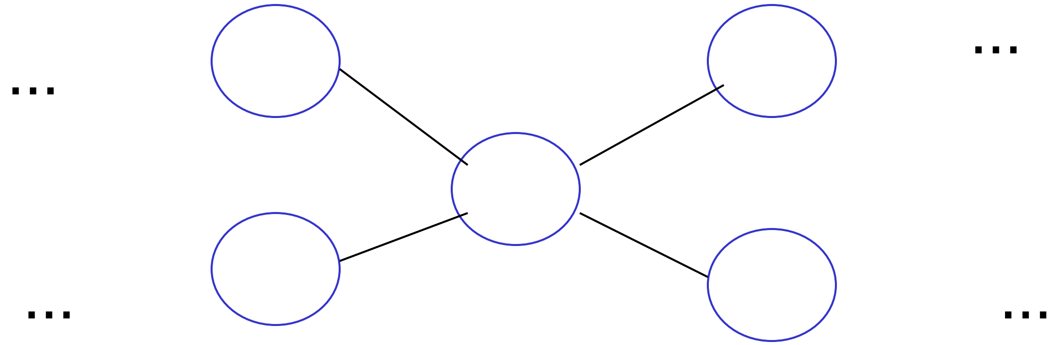
CLARA (Clustering Large Applications) (1990)

- *CLARA* (Kaufmann and Rousseeuw in 1990)
  - Built in statistical analysis packages, such as S+
- 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

CLARA (Clustering Large Applications) (1990)

- Algorithm (n=5, s = 40+2k)
  - Repeat n times:
    - Draw sample of s objects from the entire data set and perform PAM to find k mediods of the sample
    - assign each non-selected object in the entire data set to the most similar mediod
    - Calculate average dissimilarity of the clustering. If the value is smaller than the current minimum, use this value as current minimum and retain the k medoids as best so far.

Graph abstraction



Node represents k objects (medoids), a potential solution for the clustering.

Nodes are neighbors if the sets of objects differ by one object.

Each node has k(n-k) neighbors.

Cost differential between two neighbors is *TCih* (with Oi and Oh are the differing nodes in the mediod sets)

Graph Abstraction

- PAM searches for node in the graph with minimum cost
- CLARA searches in smaller graphs (as it uses PAM on samples of the entire data set)
- CLARANS
  - Searches in the original graph
  - Searches part of the graph
  - Uses the neighbors to guide the search

CLARANS (“Randomized” CLARA) *(1994)*

- *CLARANS* (A Clustering Algorithm based on Randomized Search) (Ng and Han’94)

- It is more efficient and scalable than both *PAM* and *CLARA*

CLARANS (“Randomized” CLARA) *(1994)*

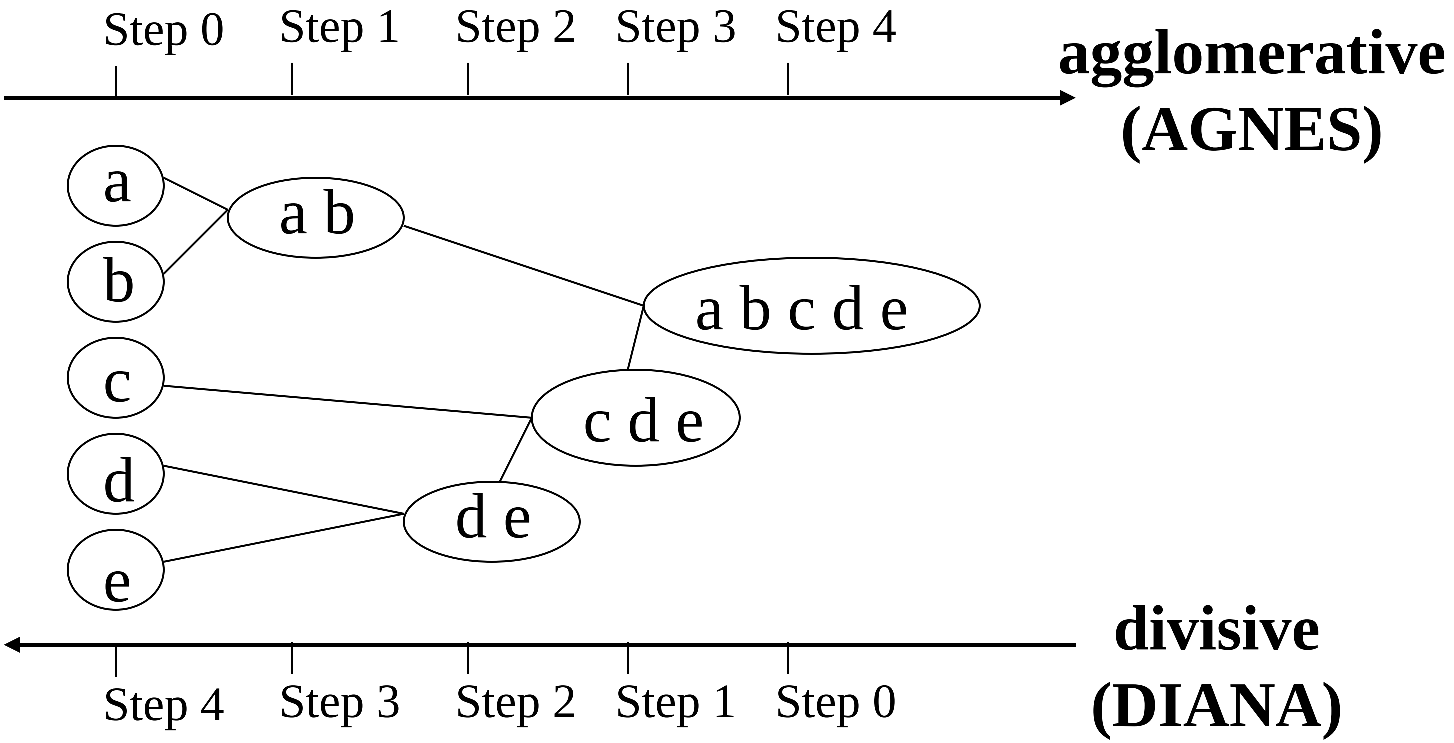
- Algorithm
  - *Numlocal*: number of local minima to be found
  - *Maxneighbor*: maximum number of neighbors to compare
  - Repeat *numlocal* times: (find local minimum)
    - Take arbitrary node in the graph
    - Consider random neighbor S of the current node and calculate the cost differential. If S has lower cost, then set S to current and repeat this step. If S does not have lower cost, repeat this step (check at most *Maxneighbor* neighbors)
    - Compare the cost of current node with minimum cost so far. If the cost is lower, set minumum cost to cost of the current node, and bestnode to the current node.

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods

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



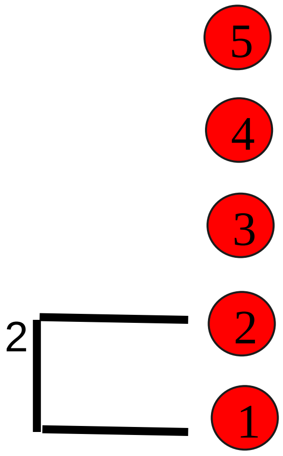
Complete-link Clustering Example

$$\begin{array}{c|ccccc|} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & & & & \\ 2 & 2 & 0 & & & \\ 3 & 6 & 3 & 0 & & \\ 4 & 10 & 9 & 7 & 0 & \\ 5 & 9 & 8 & 5 & 4 & 0 \end{array} \Rightarrow \begin{array}{c|cccc|} & (1,2) & 3 & 4 & 5 \\ \hline (1,2) & 0 & & & \\ 3 & 6 & 0 & & \\ 4 & 10 & 7 & 0 & \\ 5 & 9 & 5 & 4 & 0 \end{array}$$

$$d_{(1,2),3} = \max\{d_{1,3}, d_{2,3}\} = \max\{6,3\} = 6$$

$$d_{(1,2),4} = \max\{d_{1,4}, d_{2,4}\} = \max\{10,9\} = 10$$

$$d_{(1,2),5} = \max\{d_{1,5}, d_{2,5}\} = \max\{9,8\} = 9$$

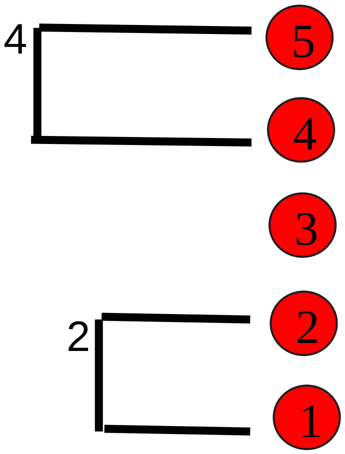


Complete-link Clustering Example

$$\begin{array}{cc} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & & & & \\ 2 & 0 & & & \\ 6 & 3 & 0 & & \\ 10 & 9 & 7 & 0 & \\ 9 & 8 & 5 & 4 & 0 \end{bmatrix} \end{array} \Rightarrow \begin{array}{cc} & \begin{array}{cccc} (1,2) & 3 & 4 & 5 \end{array} \\ \begin{array}{c} (1,2) \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & & & \\ 6 & 0 & & \\ 10 & 7 & 0 & \\ 9 & 5 & 4 & 0 \end{bmatrix} \end{array} \Rightarrow \begin{array}{cc} & \begin{array}{ccc} (1,2) & 3 & (4,5) \end{array} \\ \begin{array}{c} (1,2) \\ 3 \\ (4,5) \end{array} & \begin{bmatrix} 0 & & \\ 6 & 0 & \\ 10 & 7 & 0 \end{bmatrix} \end{array}$$

$$d_{(1,2),(4,5)} = \max\{d_{(1,2),4}, d_{(1,2),5}\} = \max\{10,9\} = 10$$

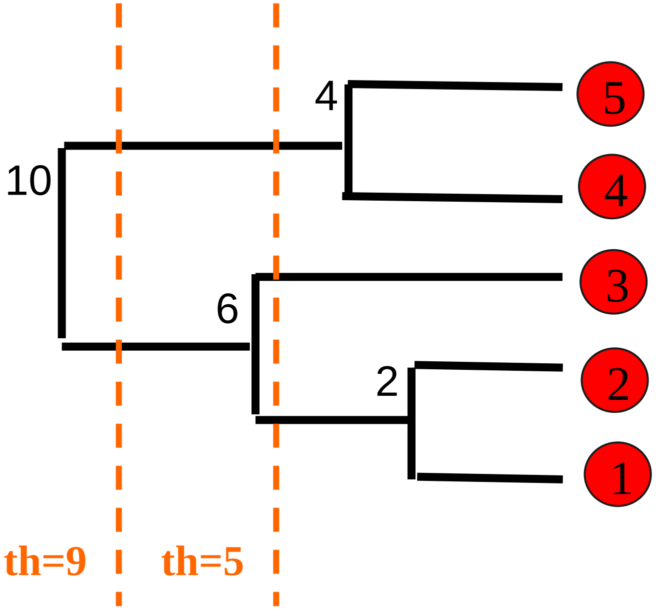
$$d_{3,(4,5)} = \max\{d_{3,4}, d_{3,5}\} = \max\{7,5\} = 7$$



Complete-link Clustering Example

$$\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & & & & \\ 2 & 2 & 0 & & & \\ 3 & 6 & 3 & 0 & & \\ 4 & 10 & 9 & 7 & 0 & \\ 5 & 9 & 8 & 5 & 4 & 0 \end{array} \Rightarrow \begin{array}{c|cccc} & (1,2) & 3 & 4 & 5 \\ \hline (1,2) & 0 & & & \\ 3 & 6 & 0 & & \\ 4 & 10 & 7 & 0 & \\ 5 & 9 & 5 & 4 & 0 \end{array} \Rightarrow \begin{array}{c|ccc} & (1,2) & 3 & (4,5) \\ \hline (1,2) & 0 & & \\ 3 & 6 & 0 & \\ (4,5) & 10 & 7 & 0 \end{array}$$

$$d_{(1,2,3),(4,5)} = \max\{ d_{(1,2),(4,5)}, d_{3,(4,5)} \} = 10$$

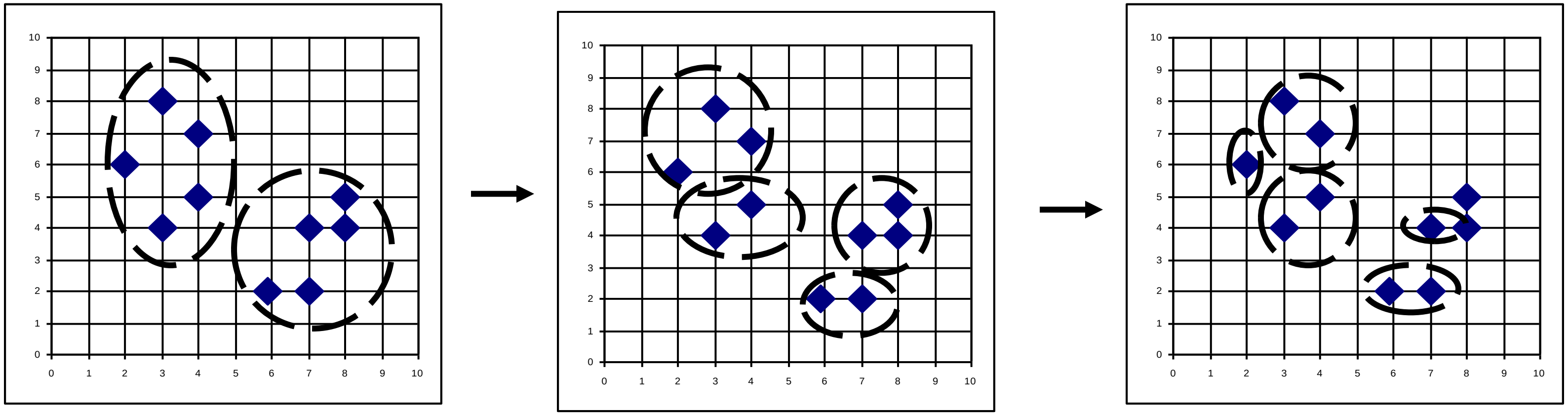


AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis 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

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



Recent Hierarchical Clustering Methods

- Major weakness of agglomerative clustering methods
  - do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
  - can never undo what was done previously
- Integration of hierarchical with distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - ROCK (1999): clustering categorical data by neighbor and link analysis
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

BIRCH (1996)

- 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, not always natural clusters.

Clustering Feature Vector in BIRCH

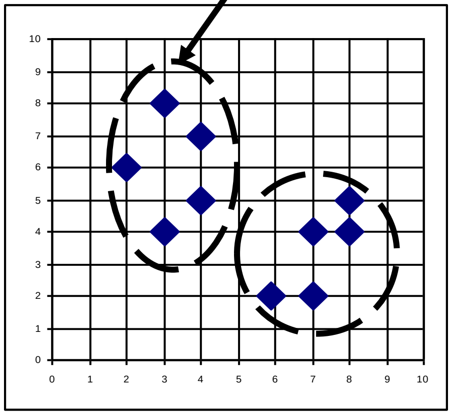
Clustering Feature: *CF = (N, LS, SS)*

N: **Number of data points**

LS: $\sum^{N}_{i=1} = \overrightarrow{X_i}$

SS: $\sum^{N}_{i=1} = \overrightarrow{X_i}^{2}$

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



(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

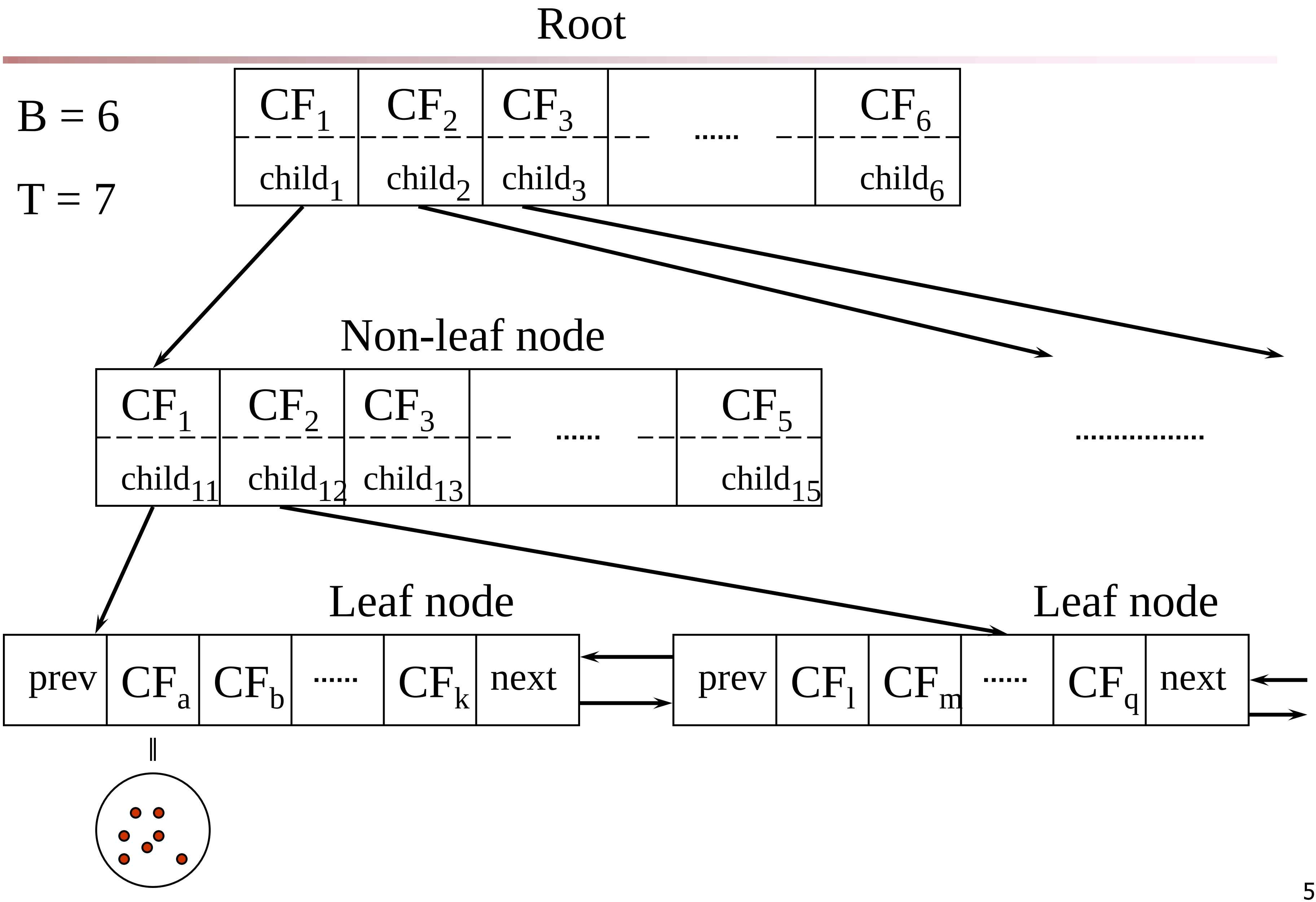
CF-Tree in BIRCH

- 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 children and stores the sums of the CFs of their children
  - A nonleaf node represents a cluster made of the subclusters represented by its children
  - A leaf node represents a cluster made of the subclusters represented by its entries
- A CF tree has two parameters
  - Branching factor: specify the maximum number of children.
  - threshold: max diameter of sub-clusters stored at the leaf nodes

The CF Tree Structure

$B = 6$

$T = 7$



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

    maximize the sum of the number of links between points within a cluster, minimize the sum of the number of links for points in different clusters
  - Computational complexity:

$$O(n^2 + nm_m m_a + n^2 \log n)$$

Similarity Measure in ROCK

- Traditional measures for *categorical data* may not work well, e.g., Jaccard coefficient
- Example: Two groups (clusters) of transactions
  - C_1. <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_2. <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}
- Jaccard coefficient may lead to wrong clustering result
  - C_1: 0.2 ({a, b, c}, {b, d, e}} to 0.5 ({a, b, c}, {a, b, d})
  - C_1 & C_2: could be as high as 0.5 ({a, b, c}, {a, b, f})
- Jaccard coefficient-based similarity function:

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

  - Ex. Let T_1 = {a, b, c}, T_2= {c, d, e}

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

Link Measure in ROCK

- Neighbor: p1 and p2 are neighbors

  iff sim(p1,p2) >= t

  (sim and t between 0 and 1)

- Link(pi,pj) is the number of common neighbors between pi and pj

Link Measure in ROCK

- Links: # of common neighbors
  - C_1 <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_2 <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}
- Let T_1 = {a, b, c}, T_2 = {c, d, e}, T_3 = {a, b, f} and sim the Jaccard coefficient similarity and t=0.5
  - link(T_1, T_2) *= 4, since they have 4 common neighbors*
    - {a, c, d}, {a, c, e}, {b, c, d}, {b, c, e}
  - link(T_1, T_3) *= 5, since they have 5 common neighbors*
    - {a, b, d}, {a, b, e}, {a, b, g}, {a, b, c}, {a, b, f}

Link Measure in ROCK

- Link(Ci,Cj) = the number of cross links between clusters Ci and Cj
- G(Ci,Cj)
  = goodness measure for merging Ci and Cj
  = Link(Ci,Cj) divided by the expected number of cross links

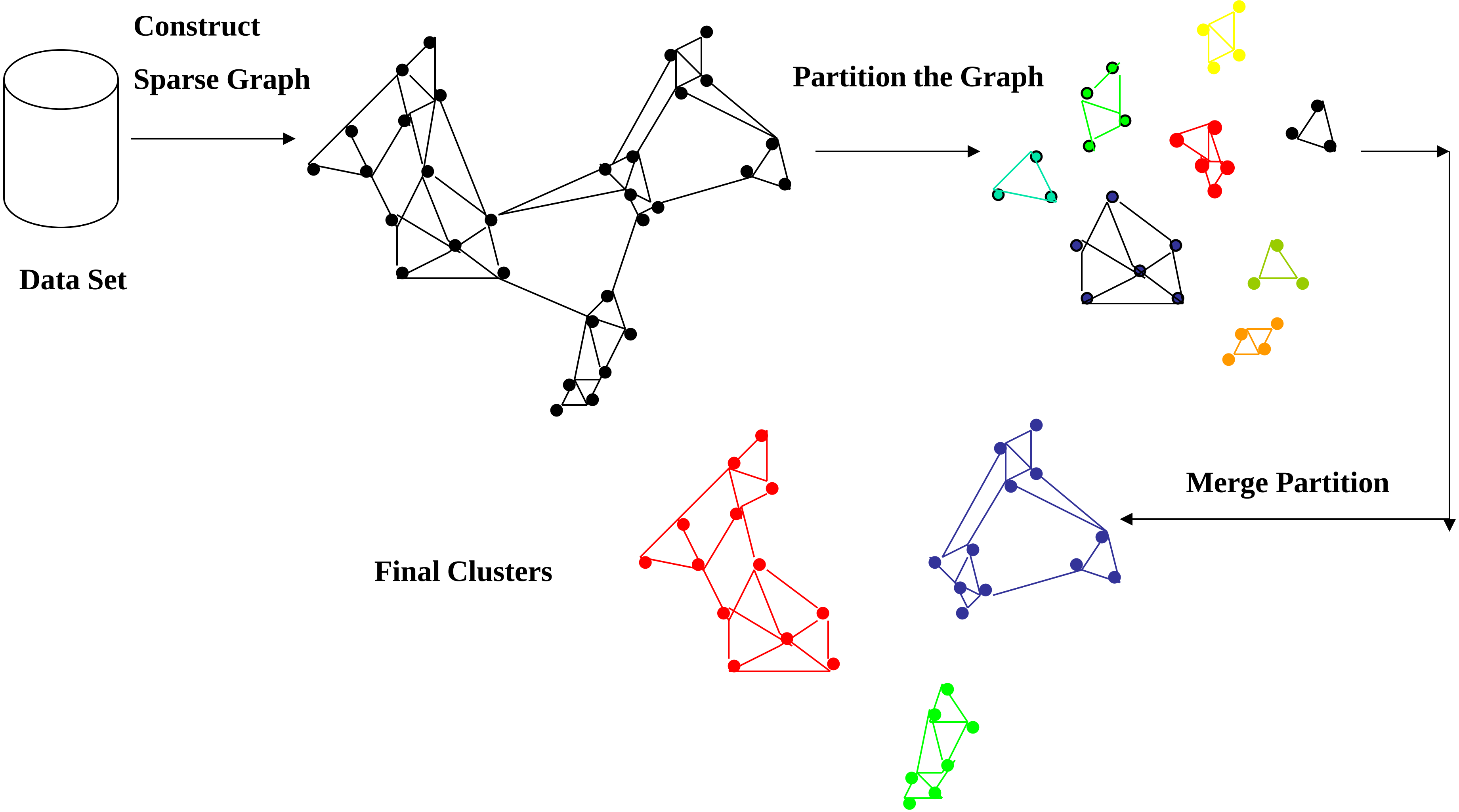
The ROCK Algorithm

- Algorithm: sampling-based clustering
  - Draw random sample
  - Hierarchical clustering with links using goodness measure of merging
  - Label data in disk: a point is assigned to the cluster for which it has the most neighbors after normalization

CHAMELEON: Hierarchical Clustering Using Dynamic Modeling (1999)

- CHAMELEON: by G. Karypis, E.H. Han, and V. Kumar'99
- 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
- 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

Overall Framework of CHAMELEON



Construct

Sparse Graph

Partition the Graph

Data Set

Merge Partition

Final Clusters

CHAMELEON: Hierarchical Clustering Using Dynamic Modeling (1999)

- A two-phase algorithm

1. Use a graph partitioning algorithm: cluster objects into a large number of relatively small sub-clusters
   - Based on k-nearest neighbor graph
   - Edge between two nodes if points corresponding to either of the nodes are among the k-most similar points of the point corresponding to the other node
   - Edge weight is density of the region
   - Dynamic notion of neighborhood: in regions with high density, a neighborhood radius is small, while in sparse regions the neighborhood radius is large
2.

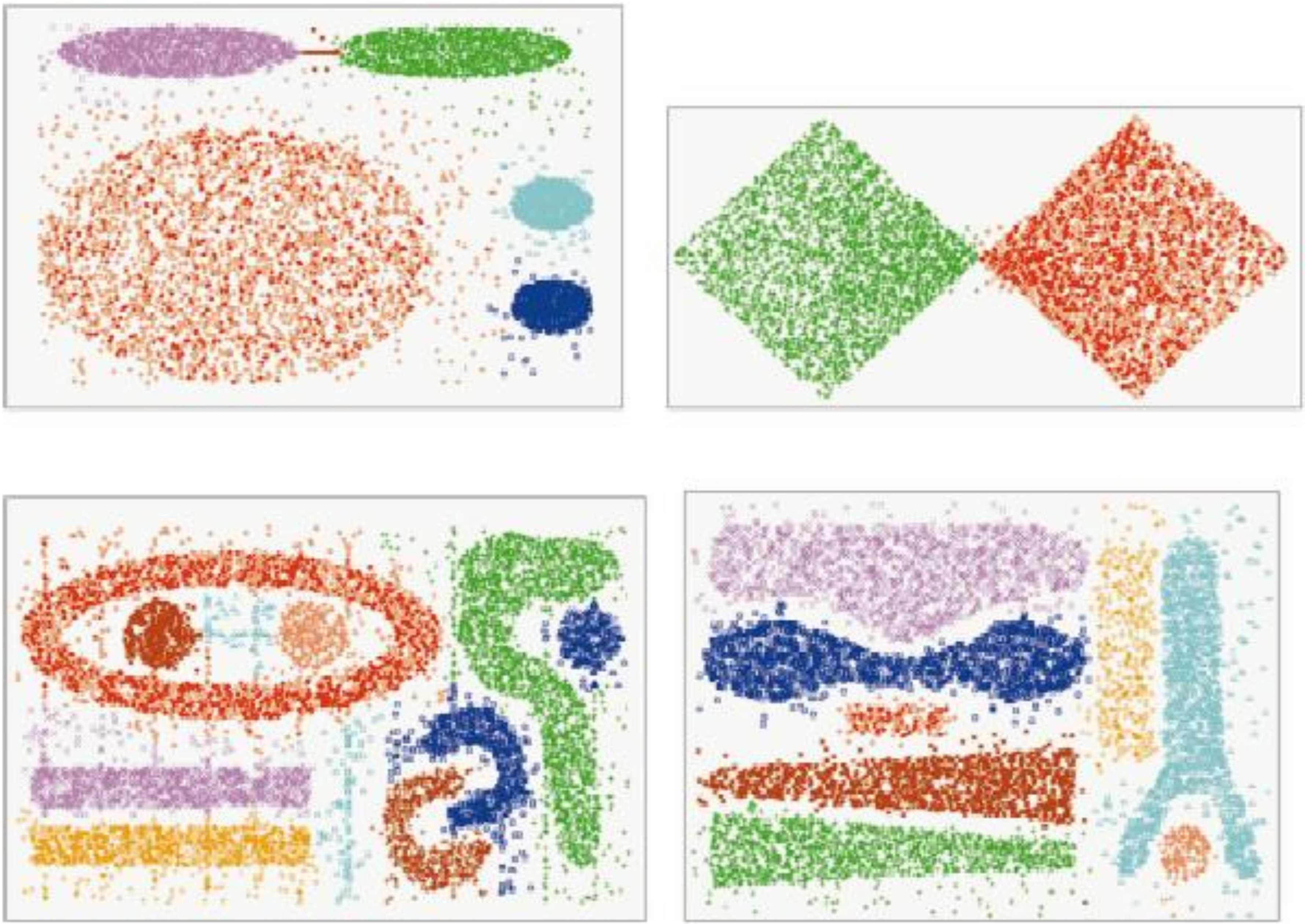
CHAMELEON: Hierarchical Clustering Using Dynamic Modeling (1999)

- A two-phase algorithm

1.

2. Use an agglomerative hierarchical clustering algorithm: find the genuine clusters by repeatedly combining these sub-clusters
    - Interconnectivity between clusters Ci and Cj: normalized sum of the weights of the edges that connect nodes in Ci and Cj
    - Closeness of clusters Ci and Cj: average similarity between points in Ci that are connected to points in Cj
    - Merge if both measures are above user-defined thresholds

CHAMELEON (Clustering Complex Objects)



Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods

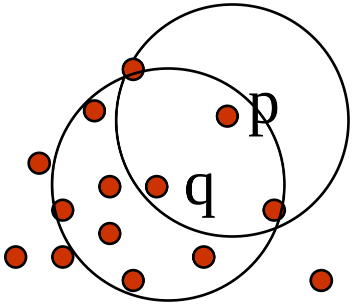
Density-Based Clustering Methods

- 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 neighborhood
  - *MinPts*: Minimum number of points in an Eps-neighborhood of that point
- $N_{Eps}(p)$: $\{q \text{ belongs to } D \mid dist(p,q) <= Eps\}$
- 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:

$$|N_{Eps}(q)| >= MinPts$$

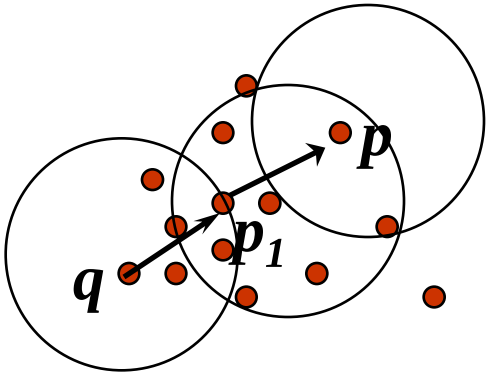


MinPts = 5

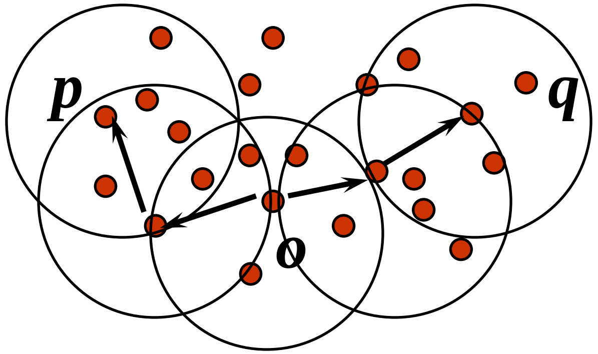
Eps = 1 cm

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 $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i

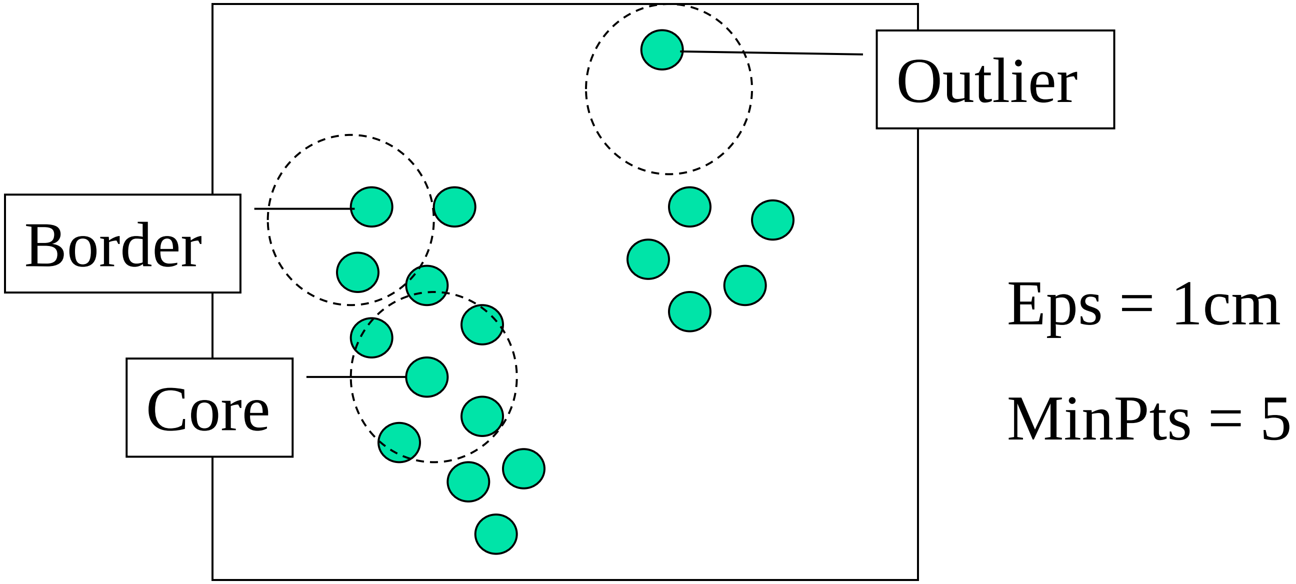


- Density-connected
  - A point *p* is density-connected to a point *q* w.r.t. *Eps*, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. *Eps* and *MinPts*



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.

DBSCAN: Sensitive to Parameters

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

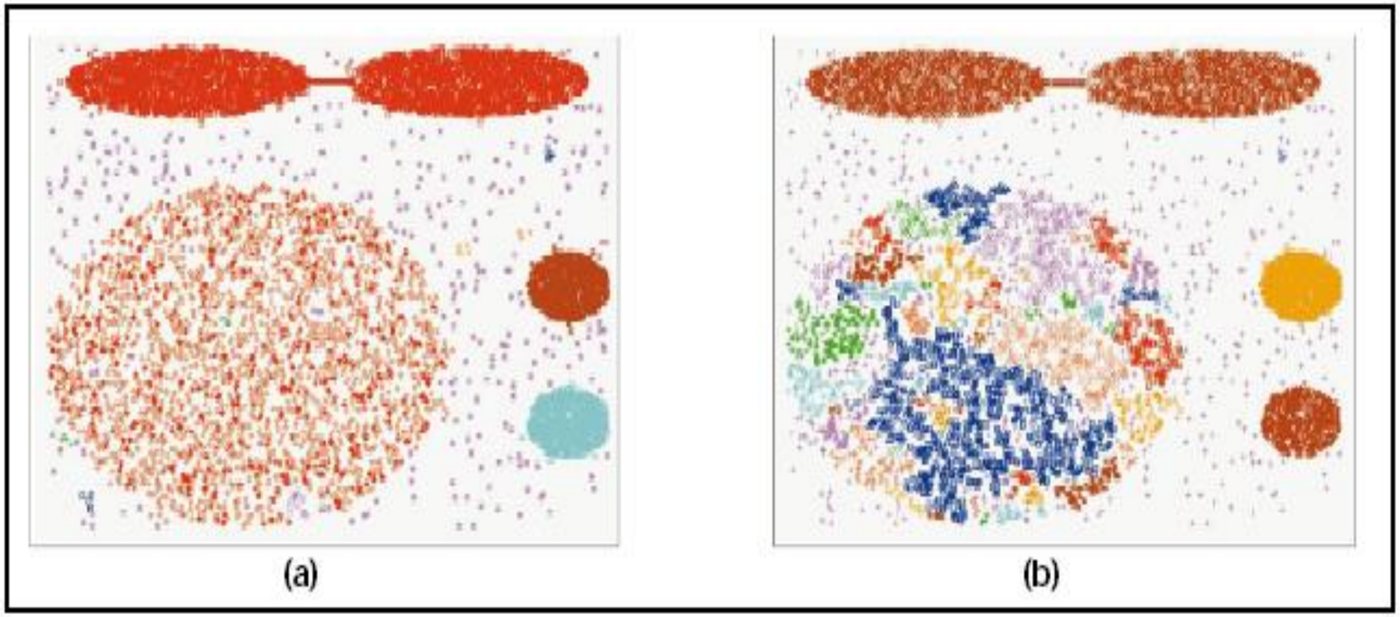
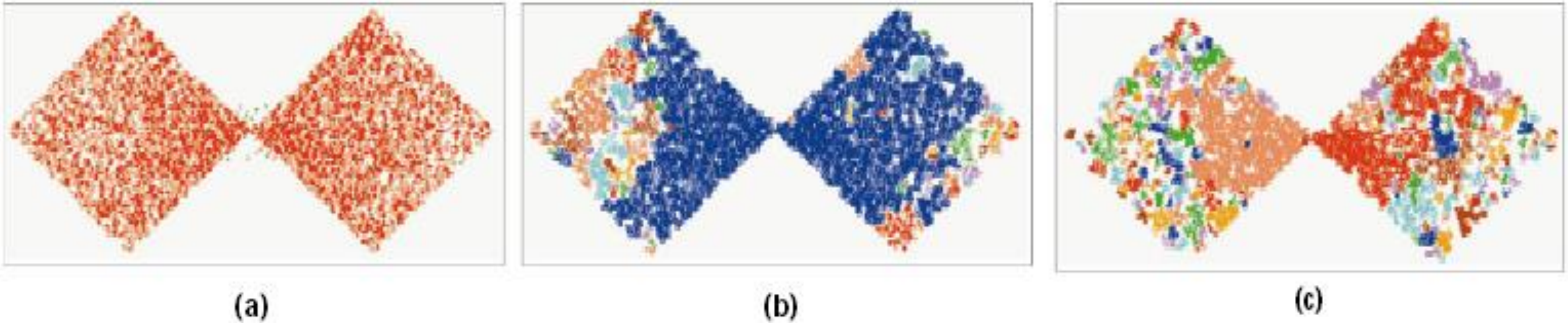
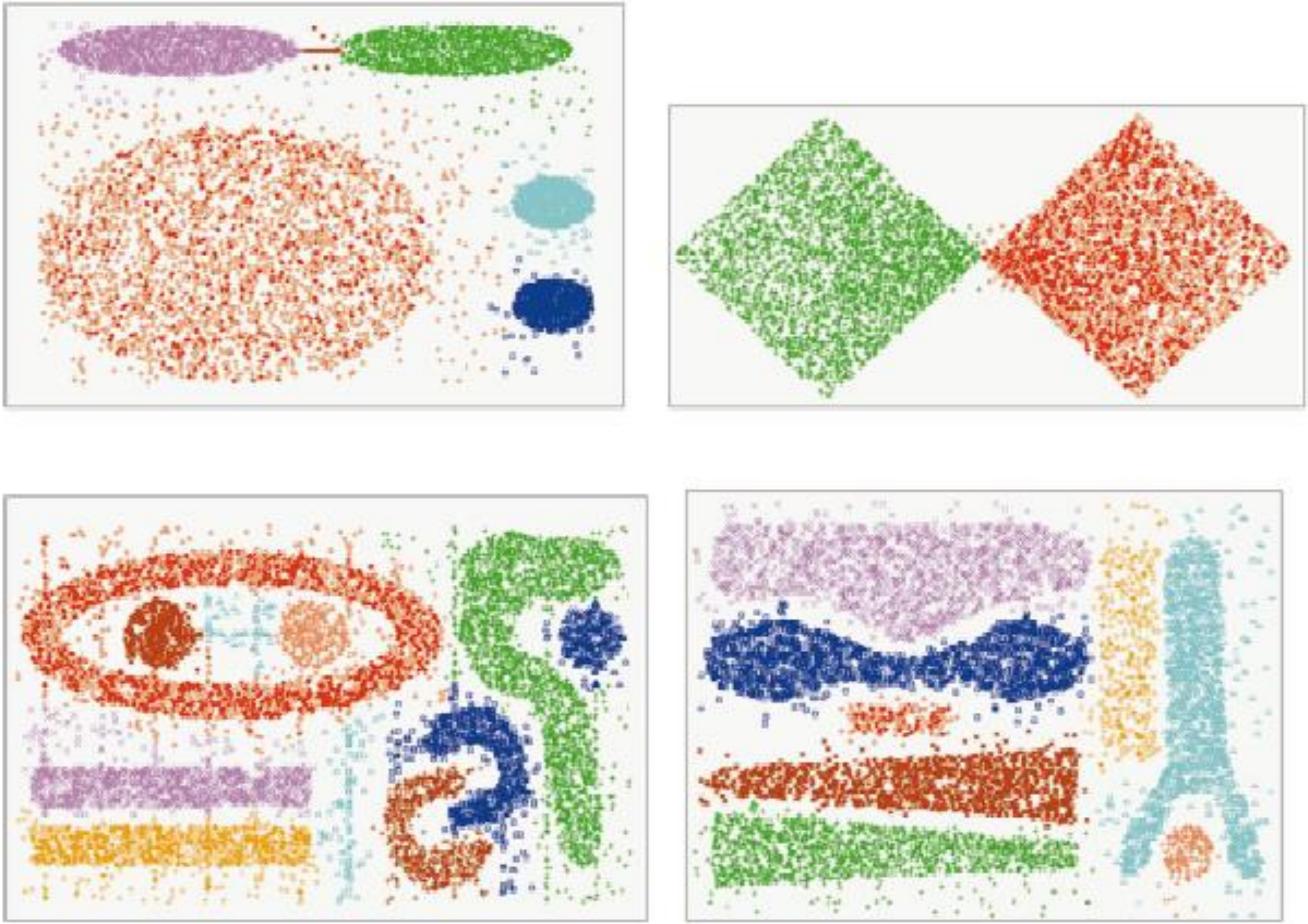


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



CHAMELEON (Clustering Complex Objects)

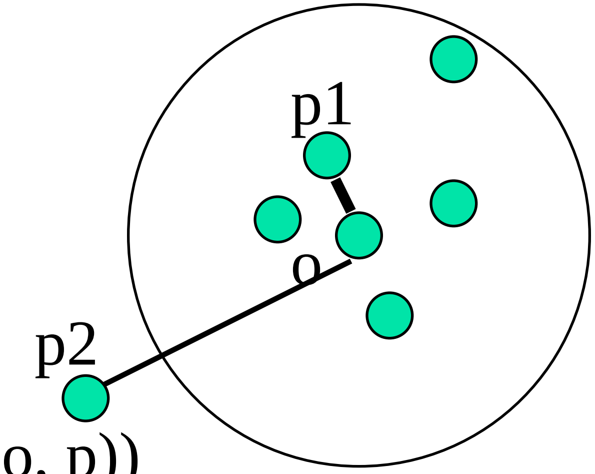


OPTICS: A Cluster-Ordering Method (1999)

- 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 equivalent 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

- Core Distance of p wrt MinPts: smallest distance eps' between p and an object in its eps-neighborhood such that p would be a core object for eps' and MinPts. Otherwise, undefined.
- Reachability Distance of p wrt o:
  Max (core-distance (o), d (o, p)) if o is core object.
  Undefined otherwise



Max (core-distance (o), d (o, p))

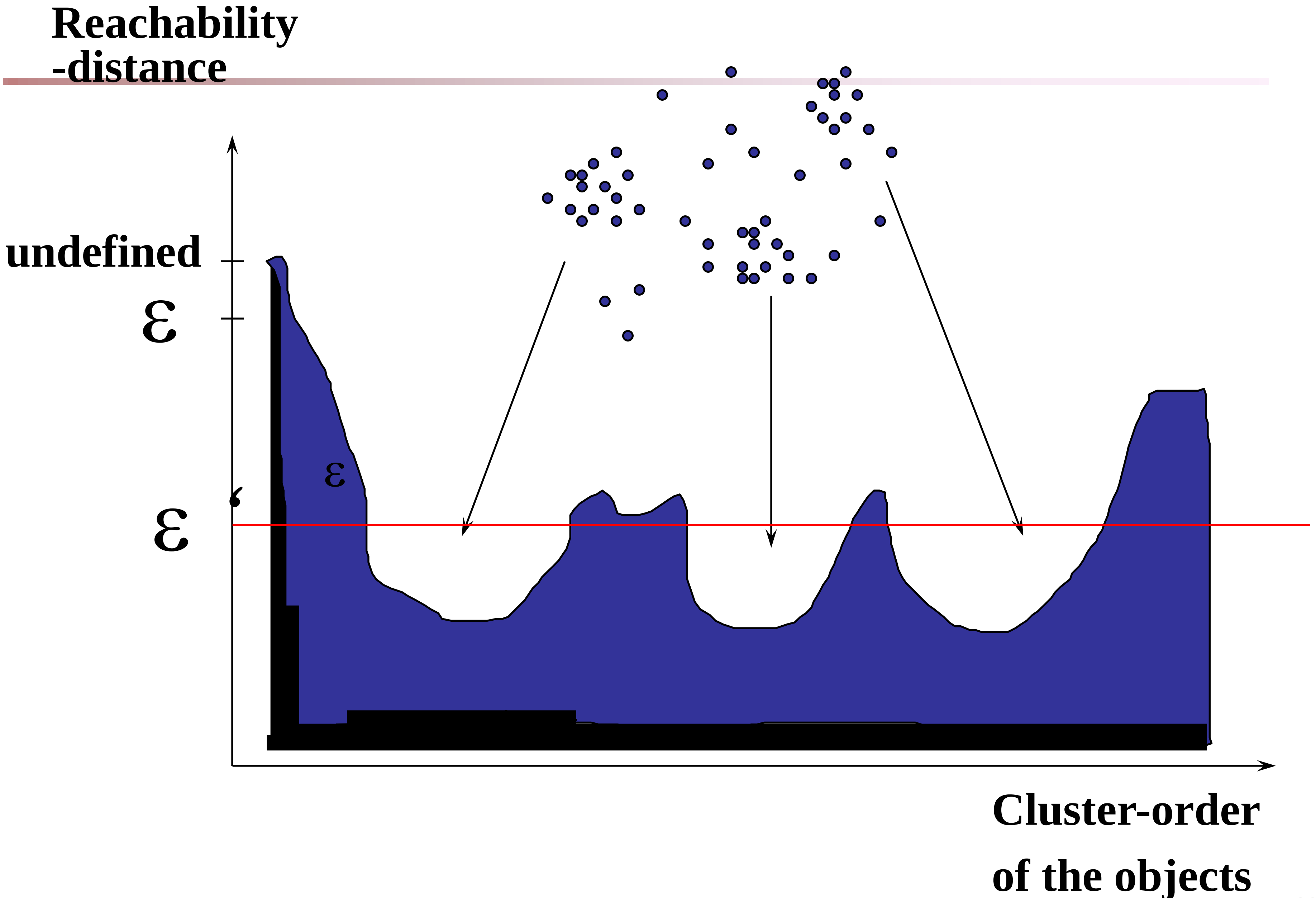
r(p1, o) = 1.5cm. r(p2,o) = 4cm

MinPts = 5

ε = 3 cm

OPTICS

- (1) Select non-processed object o
- (2) Find neighbors (eps-neighborhood)
- Compute core distance for o
- Write object o to ordered file and mark o as processed
- If o is not a core object, restart at (1)
- (o is a core object …)
- Put neighbors of o in Seedlist and order
  - If neighbor n is not yet in SeedList then add (n, reachability from o) else if reachability from o < current reachability, then update reachability + order SeedList wrt reachability
- Take new object from Seedlist with smallest reachability and restart at (2)


Reachability
-distance
undefined
ε
ε
ε'
ε
Cluster-order
of the objects

DENCLUE: Using Statistical Density Functions

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

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

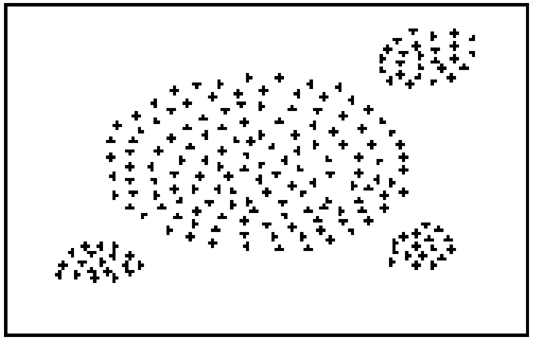
- Overall density of the data space can be calculated as the sum of the influence function of all data points

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

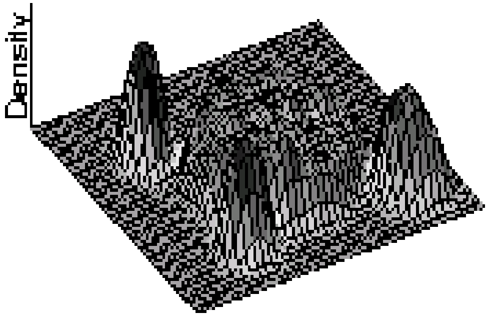
- Clusters can be determined mathematically by identifying density attractors. Density attractors are local maxima of the overall density function

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

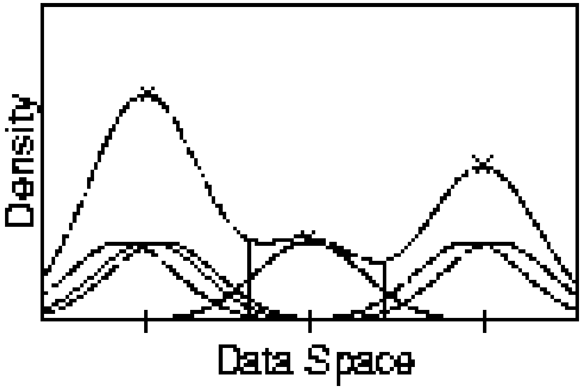
Density Attractor



(a) Data Set



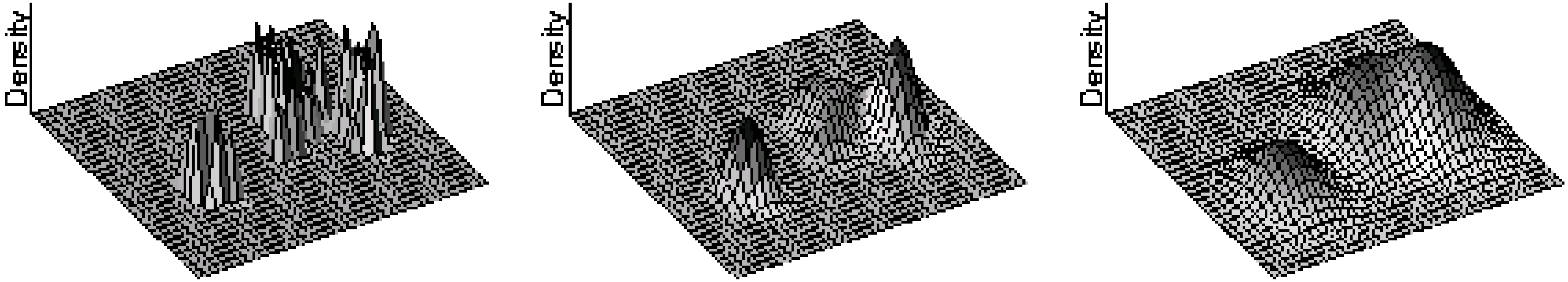
(c) Gaussian



Denclue: Technical Essence

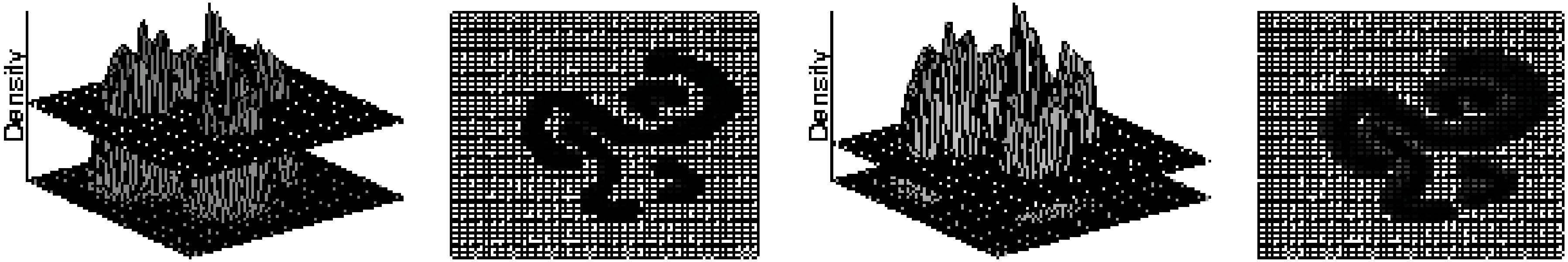
- Significant density attractor for threshold k: density attractor with density larger than or equal to k
- Set of significant density attractors X for threshold k: for each pair of density attractors x1, x2 in X there is a path from x1 to x2 such that each point on the path has density larger than or equal to k
- Center-defined cluster for a significant density attractor x for threshold k: points that are density attracted by x
  - Points that are attracted to a density attractor with density less than k are called outliers
- Arbitrary-shape cluster for a set of significant density attractors X for threshold k: points that are density attracted to some density attractor in X

Center-Defined and Arbitrary



(a) $\sigma = 0.2$ (b) $\sigma = 0.6$ (d) $\sigma = 1.5$

Figure 3: Example of Center-Defined Clusters for different σ



(a) $\xi = 2$ (b) $\xi = 2$ (c) $\xi = 1$ (d) $\xi = 1$

Figure 4: Example of Arbitray-Shape Clusters for different ξ