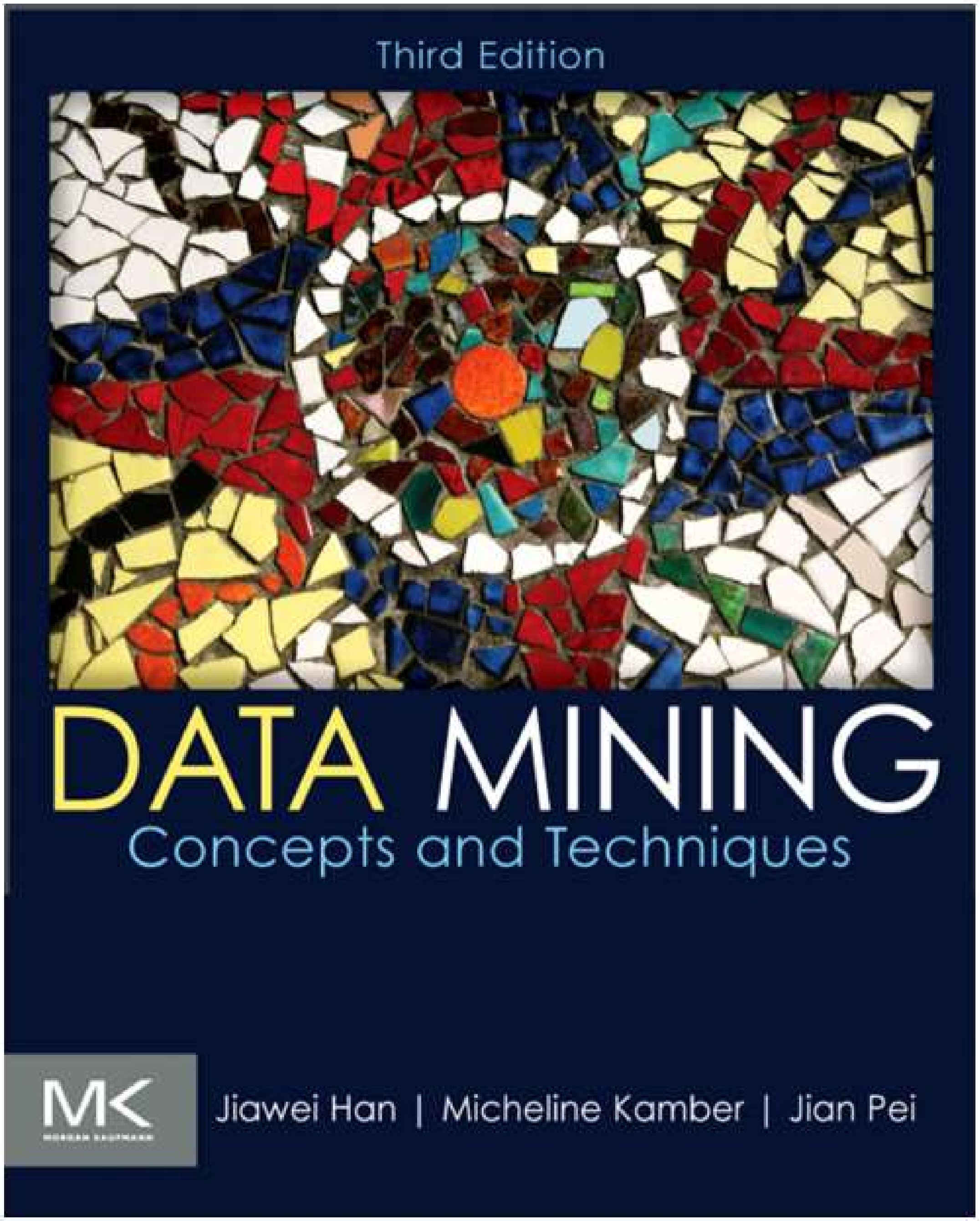


Third Edition

DATA MINING

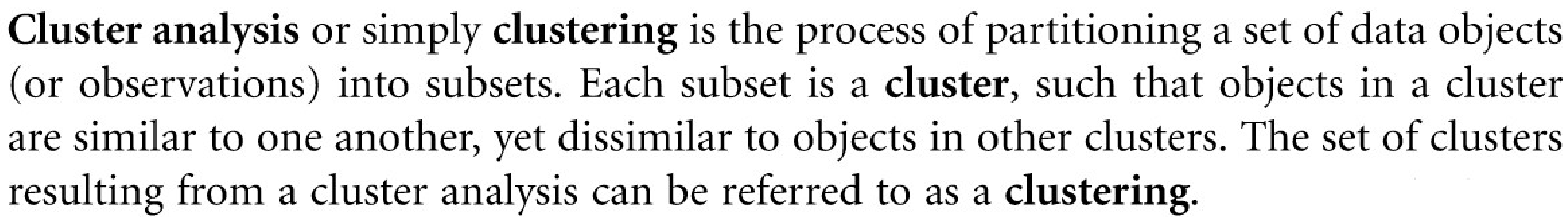
Concepts and Techniques

Jiawei Han | Micheline Kamber | Jian Pei

Module V - Cluster Analysis

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

Cluster Analysis: Basic Concepts
What is Cluster Analysis?



Cluster analysis or simply **clustering** is the process of partitioning a set of data objects (or observations) into subsets. Each subset is a **cluster**, such that objects in a cluster are similar to one another, yet dissimilar to objects in other clusters. The set of clusters resulting from a cluster analysis can be referred to as a **clustering**.

- 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)

Applications of Cluster Analysis

- Business intelligence:

  Clustering can be used to organize a large number of customers into groups, where customers within a group share strong similar characteristics.

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Web Search or Information retrieval: document clustering
- Image Recognition: Handwritten character recognition
- 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
- Security: Detection of credit card frauds.

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

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

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)

Major Clustering Approaches

- 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

Module V - Cluster Analysis

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

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), *p* object

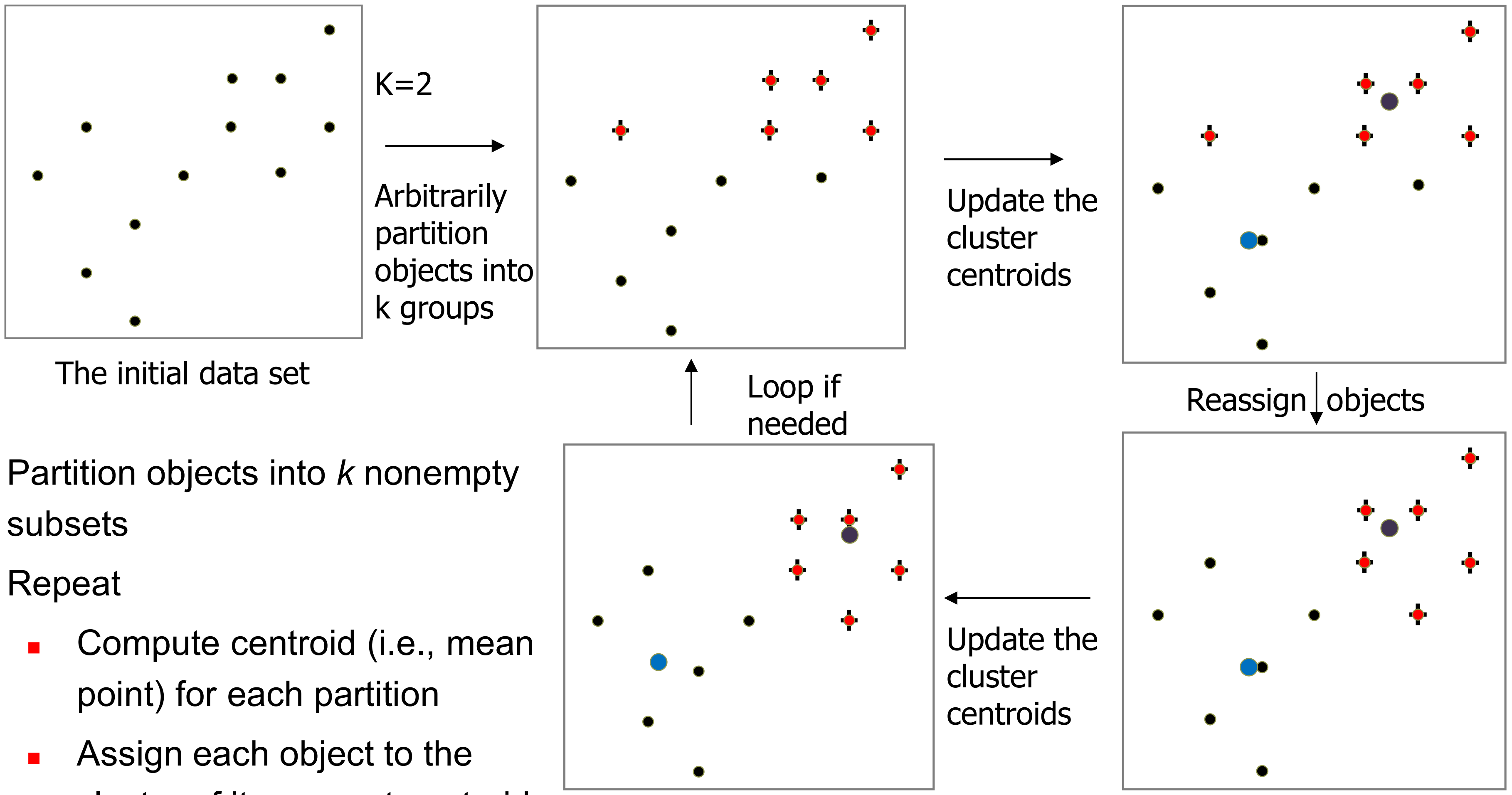
$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(\boldsymbol{p}, \boldsymbol{c_i})^2$$

- 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

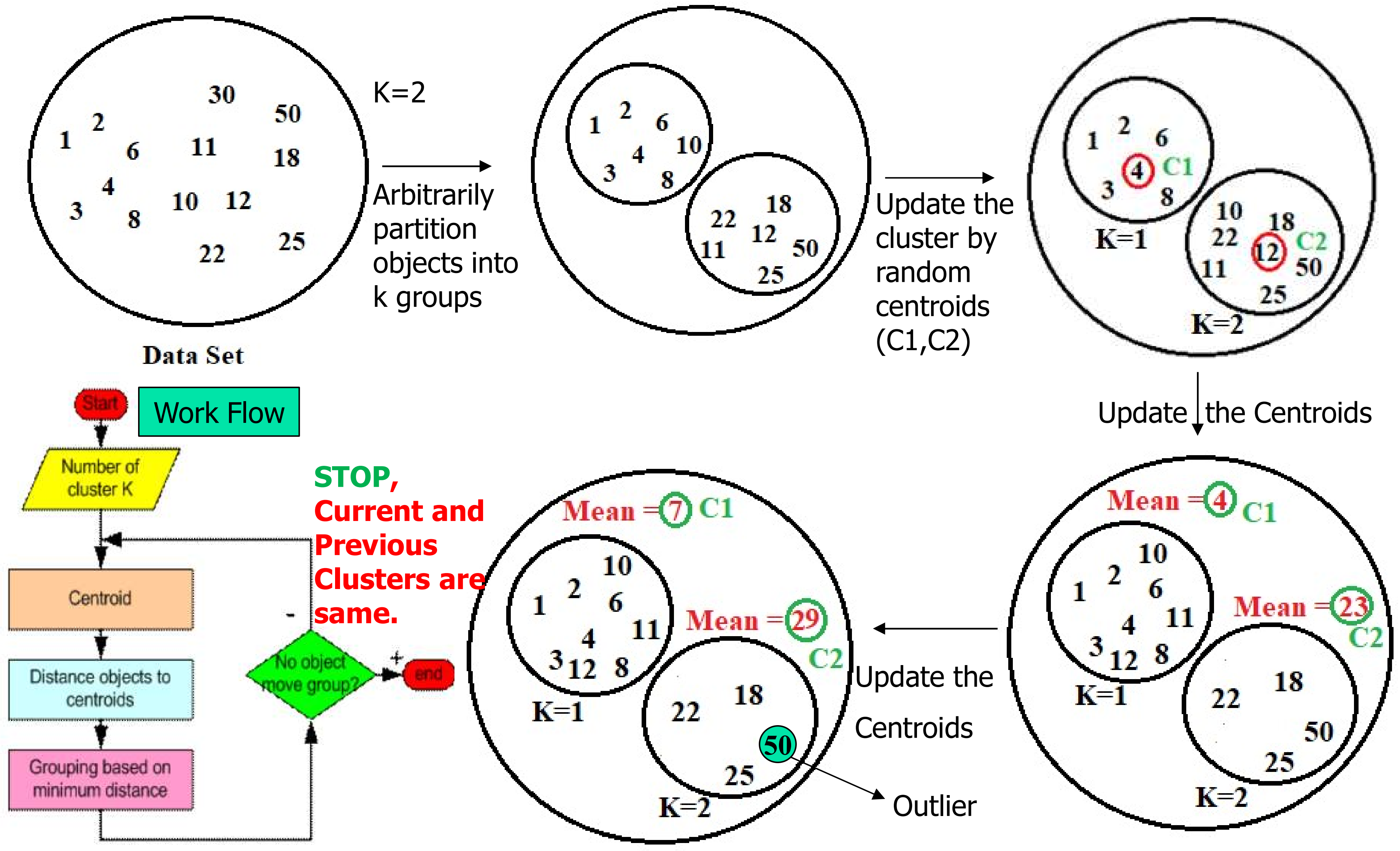
An Example of *K-Means* Clustering



K=2
Arbitrarily partition objects into k groups
The initial data set
Update the cluster centroids
Reassign objects
Update the cluster centroids
Loop if needed

- 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

K-Mean Example



Data Set
K=2
Arbitrarily partition objects into k groups
Update the cluster by random centroids (C1,C2)
C1
C2
K=1
K=2
Update the Centroids
Mean =4 C1
Mean =23 C2
Update the Centroids
Mean =7 C1
Mean =29 C2
Outlier
STOP, Current and Previous Clusters are same.
Start
Work Flow
Number of cluster K
Centroid
Distance objects to centroids
Grouping based on minimum distance
No object move group?
+
-
end

What Is the Problem of the K-Means Method?

A drawback of *k*-means. Consider six points in 1-D space having the values 1, 2, 3, 8, 9, 10, and 25, respectively. Intuitively, by visual inspection we may imagine the points partitioned into the clusters $\{1,2,3\}$ and $\{8,9,10\}$, where point 25 is excluded because it appears to be an outlier. How would *k*-means partition the values? If we apply *k*-means using $k = 2$ and Eq. (10.1), the partitioning $\{\{1,2,3\},\{8,9,10,25\}\}$ has the within-cluster variation

$$(1-2)^2 + (2-2)^2 + (3-2)^2 + (8-13)^2 + (9-13)^2 + (10-13)^2 + (25-13)^2 = 196,$$

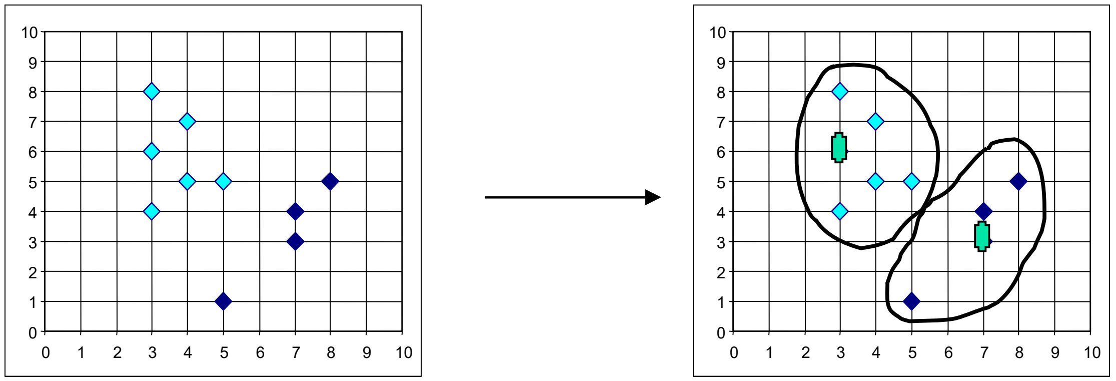
given that the mean of cluster $\{1,2,3\}$ is 2 and the mean of $\{8,9,10,25\}$ is 13. Compare this to the partitioning $\{\{1,2,3,8\},\{9,10,25\}\}$, for which *k*-means computes the within-cluster variation as

$$(1-3.5)^2 + (2-3.5)^2 + (3-3.5)^2 + (8-3.5)^2 + (9-14.67)^2 \\ + (10-14.67)^2 + (25-14.67)^2 = 189.67,$$

What Is the Problem of the K-Means Method?

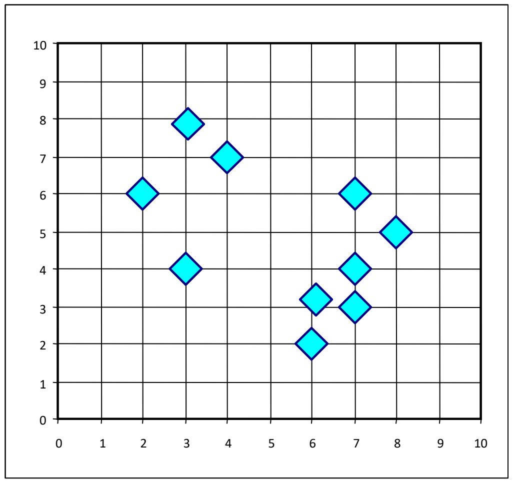
- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally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

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(\mathbf{p}, \mathbf{o}_i)$$



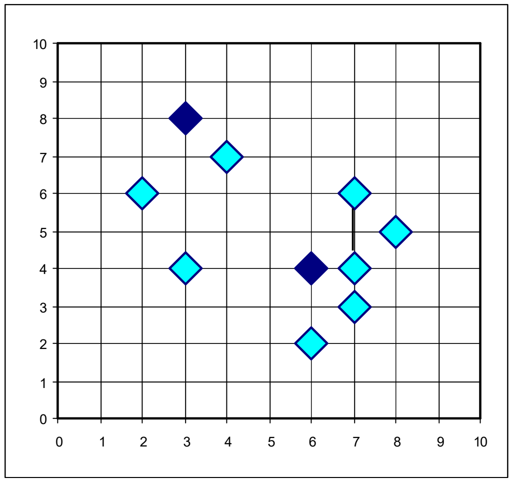
PAM: A Typical K-Medoids Algorithm

Total Cost = 20

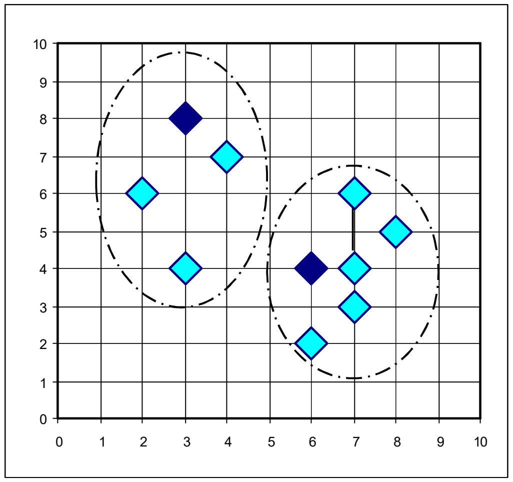


K=2

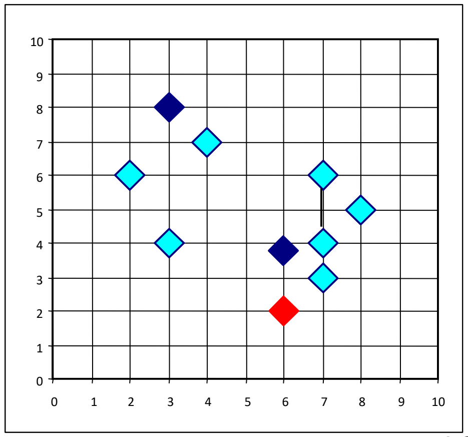
Arbitrary choose k object as initial medoids



Assign each remaining object to nearest medoids

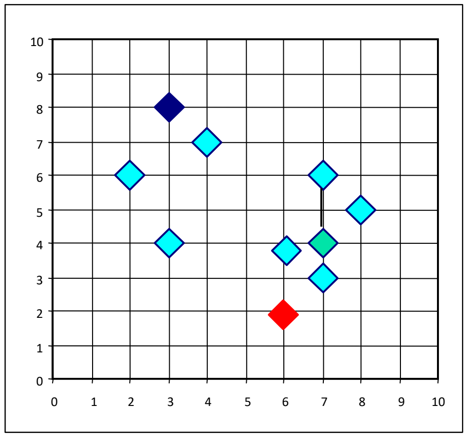


Randomly select a nonmedoid object,O_{ramdom}



Compute total cost of swapping

Total Cost = 26



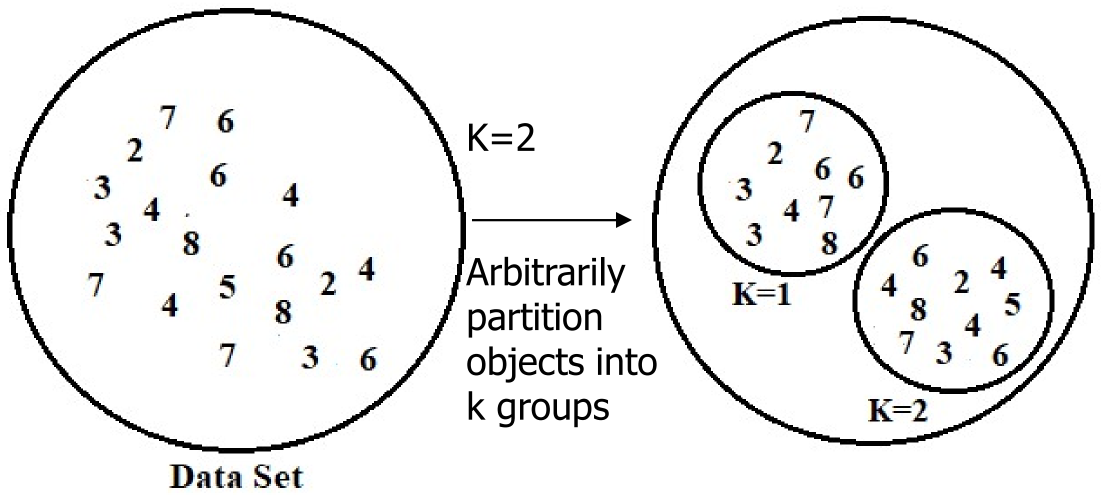
Swapping O and O_{ramdom}

If quality is improved.

Do loop

Until no change

K-Medoids Example



K=1	K=2
3	4
3	8
4	7
6	2
6	4
7	3
7	4
8	5
7	6
2	6

Arbitrary choose k object as initial medoids

K=1	K=2
3	4
3	8
4	7
6	2
6	4
7	3
7	4
8	5
7	6
2	6

Compare the difference between medoids with rest of the objects using : $|x1 - y1| + |x2 - y2|$ to calculate Cost. (3,4) and (3,8) = 0 +4 = 4

Similarly, Medoid1(3,4) ⇨ **0**, **4**, **4**, 5, 3, 5, 4, 6, 6, **3**

Medoid2(7,4) ⇨ 4, 8, 6, **3**, **1**, **1**, **0**, **2**, **2**, 7

Select minimum value from Medoid1 and Medoid2, then compute the total cost.

Total Cost : 0+4+4+3+1+1+0+2+2+3 = 20

Replace current medoid (7,4) with other medoid (8,5) and evaluate cost. Repeat, until no change in total cost.

The K-Medoid Clustering Method

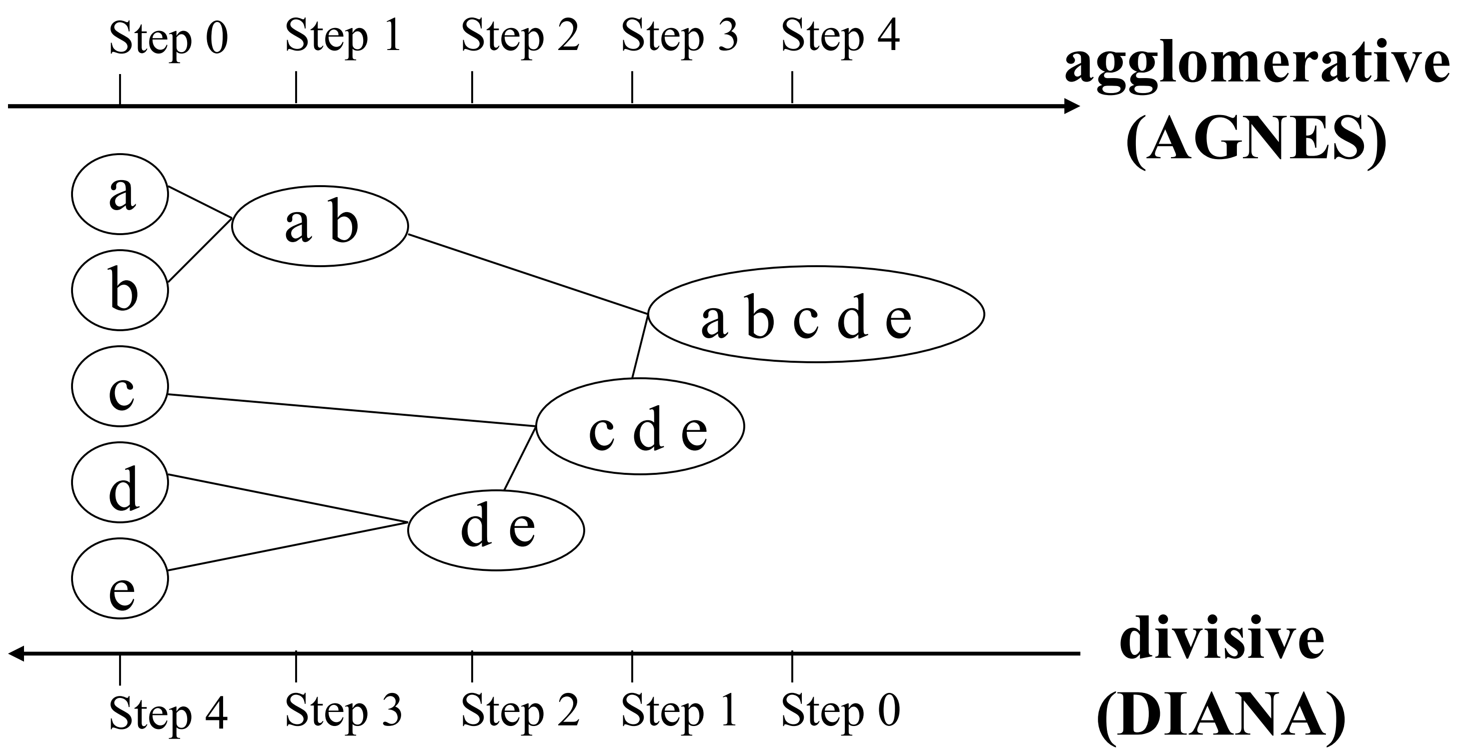
- *K-Medoids* Clustering: Find *representative* objects (<u>medoids</u>) in clusters
  - *PAM* (Partitioning Around Medoids, Kaufmann & Rousseeuw 1987)
    - Starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of 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 (Clustering LARge Applications)* (Kaufmann & Rousseeuw, 1990): PAM on samples
  - *CLA**RANS*** (Ng & Han, 1994): Randomized (search) re-sampling

Chapter 10. Cluster Analysis: Basic Concepts and Methods

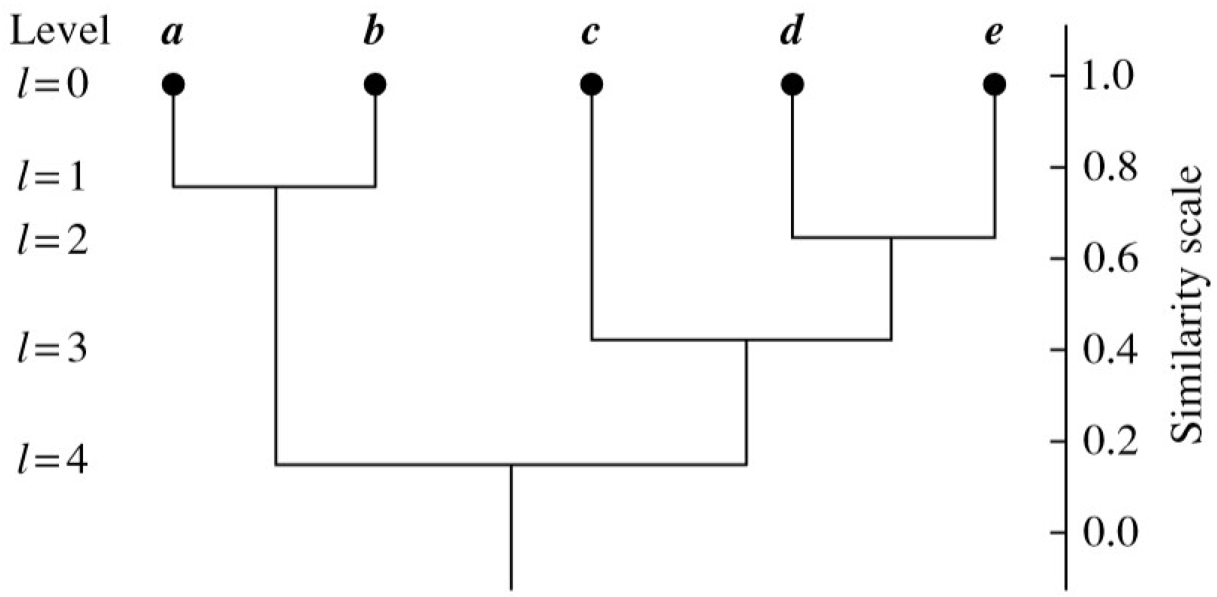
- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering

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



Dendrogram: Shows How Clusters are Merged



Dendrogram representation for hierarchical clustering of data objects $\{a, b, c, d, e\}$

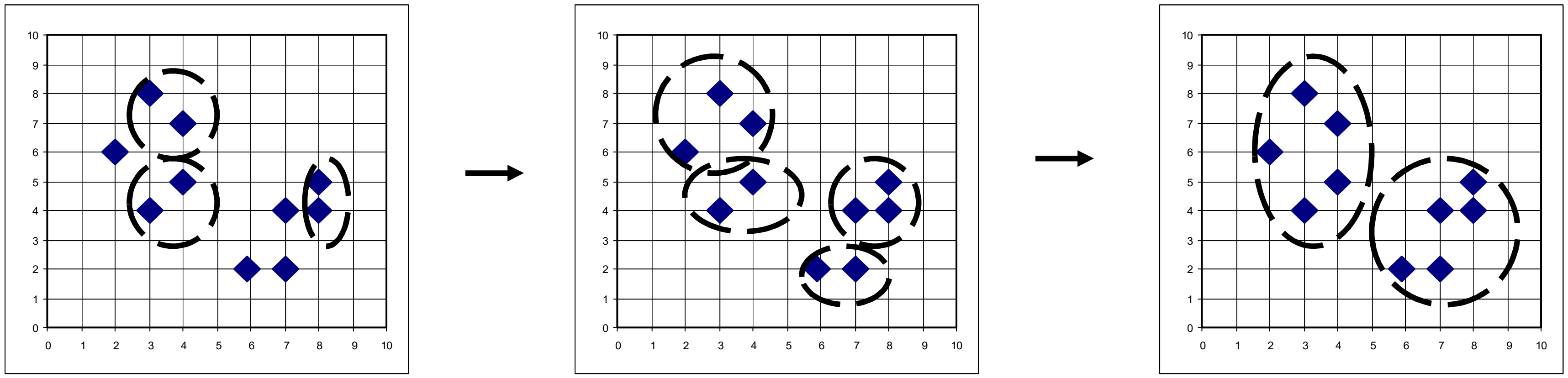
What is Dendrogram?

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

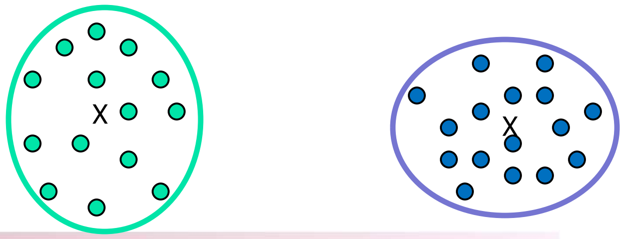
A <u>clustering</u> of the data objects is obtained by <u>cutting</u> the dendrogram at the desired level, then each <u>connected component</u> forms a cluster.

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



Distance between Clusters



Four widely used measures for distance between clusters are as follows, where $\mathbf{j}p - p^0\mathbf{j}$ is the distance between two objects or points, p and p^0; m_i is the mean for cluster, C_i; and n_i is the number of objects in C_i. They are also known as *linkage measures.*

Minimum distance: $dist_{min}(C_i, C_j) = \min_{p \in C_i, p^0 \in C_j} \{|p - p^0|\}$

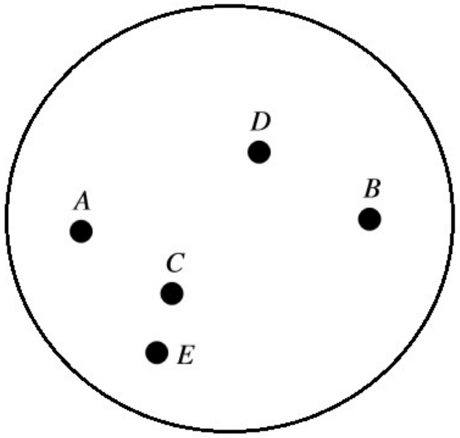
Maximum distance: $dist_{max}(C_i, C_j) = \max_{p \in C_i, p^0 \in C_j} \{|p - p^0|\}$

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

Average distance: $dist_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i, p^0 \in C_j} |p - p^0|$

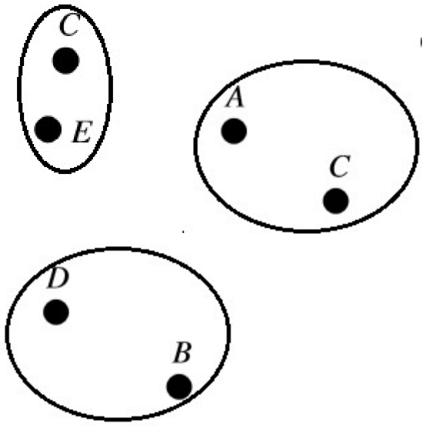
Clustering using Single Linkage

single-linkage approach ⇨ in **single-linkage** approach each cluster is represented by all the objects in the cluster, and the similarity between two clusters is measured by the similarity of the *closest* pair of data points belonging to different clusters. The cluster-merging process repeats until all the objects are eventually merged to form one cluster. This method also known as minimal spanning tree.



Data Set

	A	B	C	D	E
A	0				
B	10	0			
C	4	8	0		
D	7	5	9	0	
E	11	12	2	6	0



Sub Clusters

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

Select Minimum scale unit i.e., closest pair from the data set. [C,E] is a closest pair with scale unit 2. So merge [C,E] and update the cluster.

Clustering using Single Linkage

Dis(A,[C,E]) ➡ min(d(A,C), d(A,E))
➡ min(d(4,11)
➡ 4

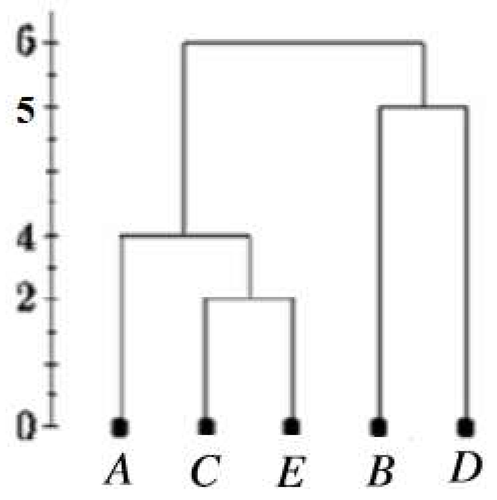
Dis(B,[C,E]) ➡ min(d(B,C), d(B,E))
➡ min(d(8,12)
➡ 8

Dis(D,[C,E]) ➡ min(d(D,C), d(D,E))
➡ min(d(9,6) ➡ 6

	A	B	[C,E]	D
A	0			
B	10	0		
[C,E]	4	8	0	
D	7	5	6	0

	[A,C,E]	B	D
[A,C,E]	0		
B	8	0	
D	6	5	0

	[A,C,E]	[B,D]
[A,C,E]	0	
[B,D]	6	0



Repeat until all elements ends up being in the same cluster.

Clustering using Complete Linkage

- The cluster are sequentially combined into larger clusters until all elements end up being in the same cluster. This method is also known as farthest neighbour clustering.

Maximum distance: $dist_{max}(C_i, C_j) = \max_{p \in C_i, p^0 \in C_j} \{|p - p^0|\}$

	A	B	C	D	E
A	0				
B	10	0			
C	4	8	0		
D	7	5	9	0	
E	11	12	2	6	0

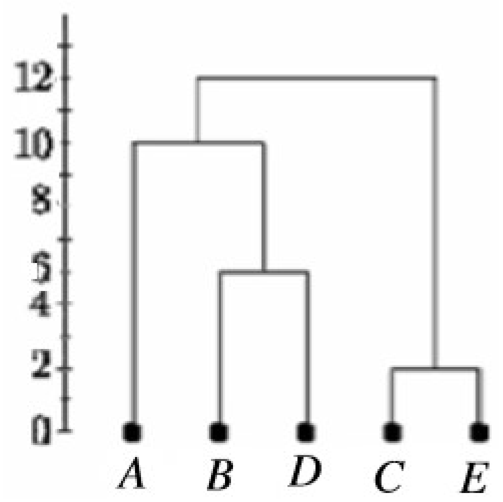
Select Minimum scale unit i.e., closest pair from the data set. [C,E] is a closest pair with scale unit 2. So merge [C,E] and update the cluster.

Clustering using Complete Linkage

Dis(A,[C,E]) $\Rightarrow$ Max(d(A,C), d(A,E))
$\Rightarrow$ Max(d(4,11)
$\Rightarrow$ 11

Dis(B,[C,E]) $\Rightarrow$ Max(d(B,C), d(B,E))
$\Rightarrow$ Max(d(8,12)
$\Rightarrow$ 12

Dis(D,[C,E]) $\Rightarrow$ Max(d(D,C), d(D,E))
$\Rightarrow$ Max(d(9,6) $\Rightarrow$ 9



Repeat until all elements ends up being in the same cluster.

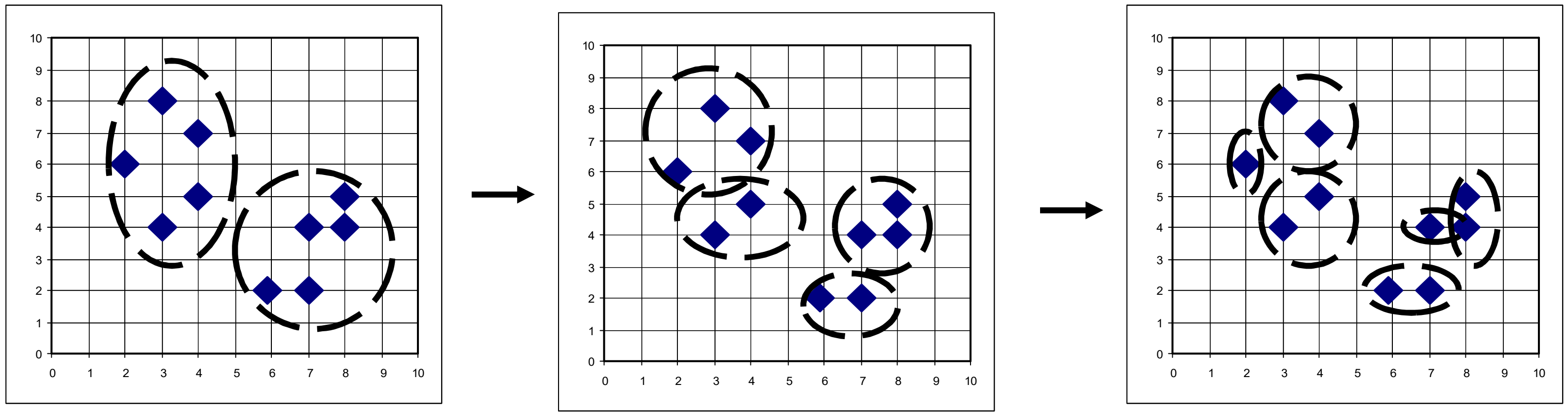
	A	B	[C,E]	D
A	0			
B	10	0		
[C,E]	11	12	0	
D	7	5	9	0

	A	[B,D]	[C,E]
A	0		
[B,D]	10	0	
[C,E]	11	12	0

	[A,B,D]	[C,E]
[A,B,D]	0	
[C,E]	12	0

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



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** is designed for clustering a large amount of numeric data.
- **Micro-Clustering:** Integrating hierarchical clustering at initial stage.
- **Macro-Clustering:** Iterative partitioning at last stage.
- **BIRCH** overcomes the two difficulties:
- (1) Scalability (2) In-ability

BIRCH

BIRCH uses the notions of *clustering feature* to summarize a cluster, and *clustering feature tree* (*CF-tree*) to represent a cluster hierarchy.

Consider a cluster of n d-dimensional data objects or points. The **clustering feature** (**CF**) of the cluster is a 3-D vector summarizing information about clusters of objects.

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

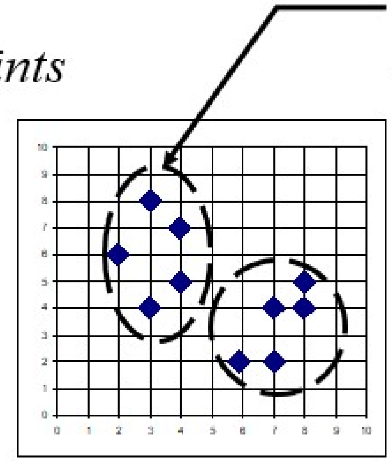
N: **Number of data points**

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)

BIRCH (Cal. the dis. b/w obj. in a Cluster)

- Centroid: the "middle" of a cluster

$$x_0 = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{LS}{n},$$

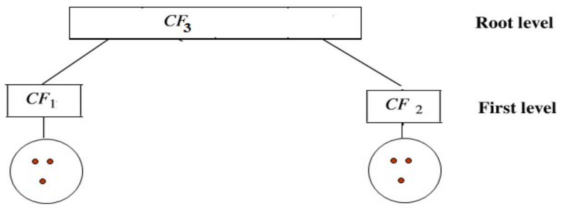
- Radius: square root of average distance from any point of the cluster to its centroid

$$R = \sqrt{\frac{\sum_{i=1}^{n}(x_i - x_0)^2}{n}} = \sqrt{\frac{nSS - 2LS^2 + nLS}{n^2}},$$

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

$$D = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i - x_j)^2}{n(n-1)}} = \sqrt{\frac{2nSS - 2LS^2}{n(n-1)}}.$$

BIRCH Example



Summarizing a cluster using the clustering feature can avoid storing the detailed information about individual objects or points. Instead, we only need a constant size of space to store the clustering feature. This is the key to BIRCH efficiency in space.

Moreover, clustering features are *additive*. That is, for two disjoint clusters, C_1 and C_2, with the clustering features $CF_1 = \langle n_1, LS_1, SS_1 \rangle$ and $CF_2 = \langle n_2, LS_2, SS_2 \rangle$, respectively, the clustering feature for the cluster that formed by merging C_1 and C_2 is simply

$$CF_1 + CF_2 = \langle n_1 + n_2, LS_1 + LS_2, SS_1 + SS_2 \rangle.$$

Clustering feature. Suppose there are three points, $(2,5), (3,2)$, and $(4,3)$, in a cluster, C_1. The clustering feature of C_1 is

$$CF_1 = \langle 3, (2+3+4, 5+2+3), (2^2+3^2+4^2, 5^2+2^2+3^2) \rangle = \langle 3, (9,10), (29,38) \rangle.$$

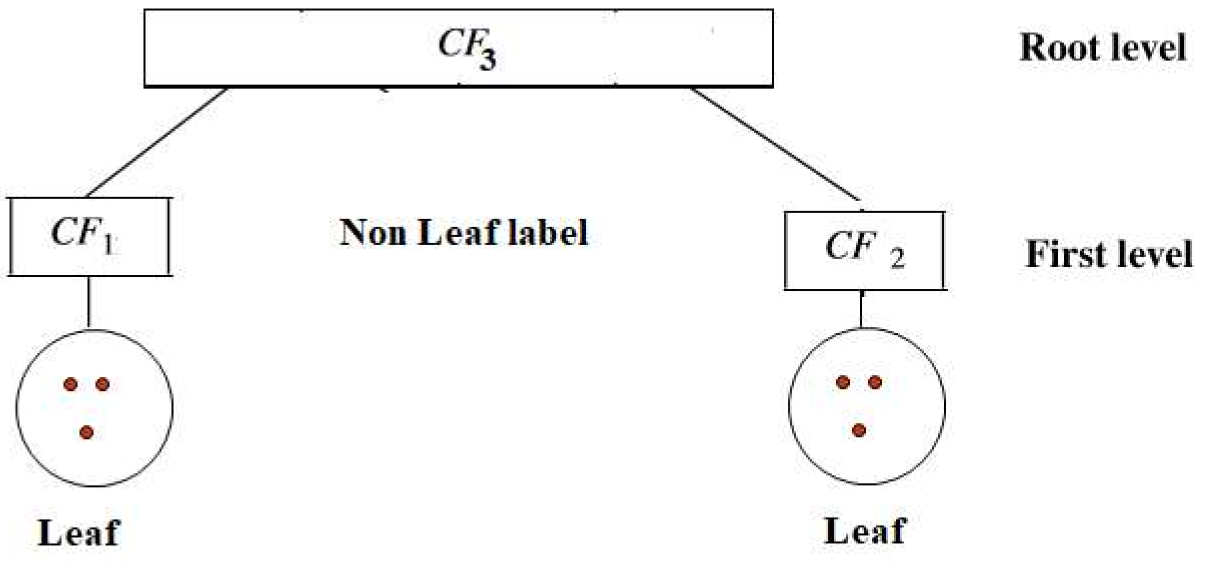
Suppose that C_1 is disjoint to a second cluster, C_2, where $CF_2 = \langle 3, (35,36), (417,440) \rangle$. The clustering feature of a new cluster, C_3, that is formed by merging C_1 and C_2, is derived by adding CF_1 and CF_2. That is,

$$CF_3 = \langle 3+3, (9+35, 10+36), (29+417, 38+440) \rangle = \langle 6, (44,46), (446,478) \rangle.$$

BIRCH Example CF-Tree

Branching factor = 1

Threshold = 3



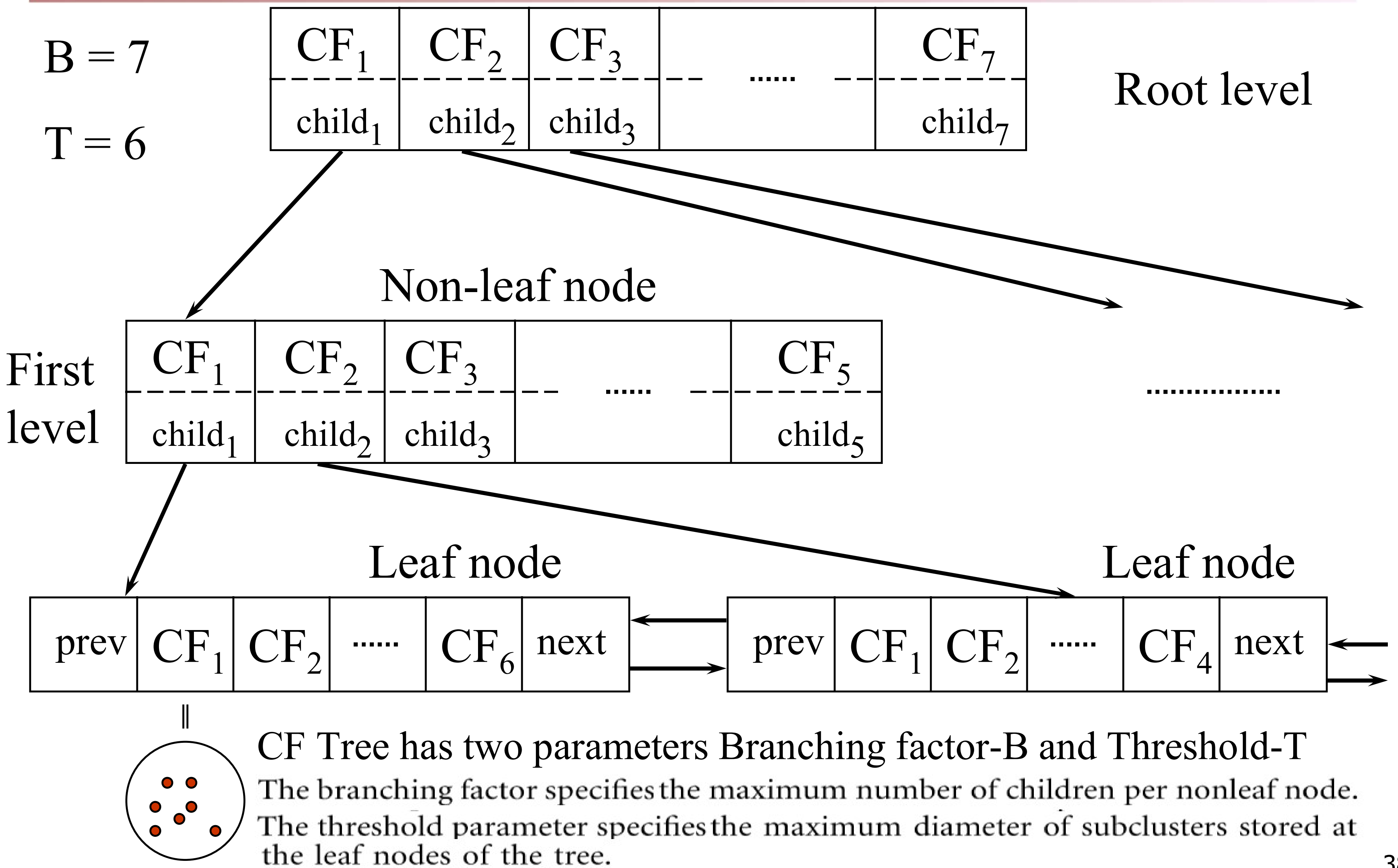
CF-Tree in BIRCH

- 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

BIRCH

- 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

The CF Tree Structure



CF Tree has two parameters Branching factor-B and Threshold-T

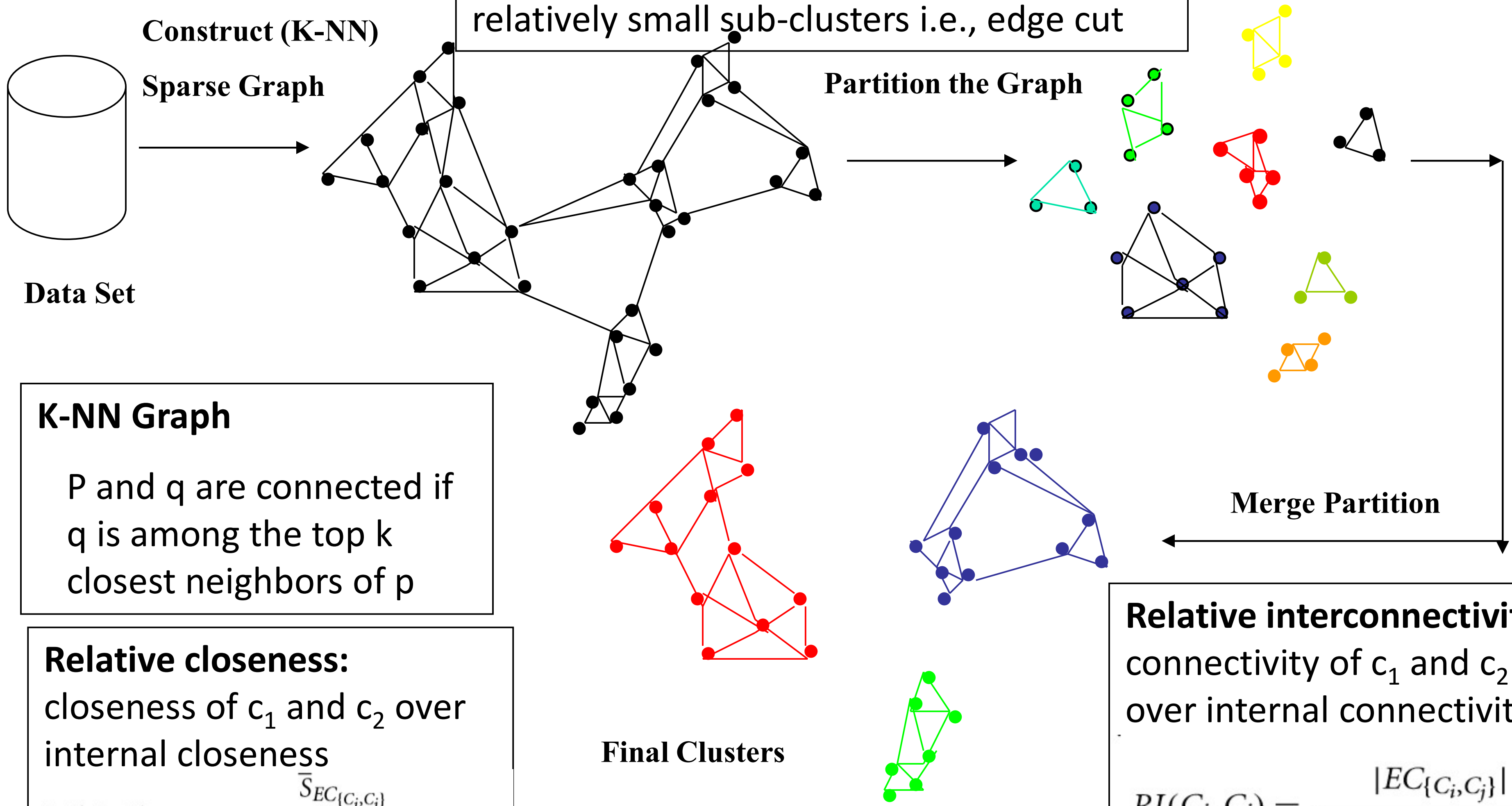
The branching factor specifies the maximum number of children per nonleaf node. The threshold parameter specifies the maximum diameter of subclusters stored at the leaf nodes of the tree.

CHAMELEON: Multiphase Hierarchical Clustering Using Dynamic Modeling

- 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
- CHAMELEON is a Graph-based model.
- It consists of 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

Partition K-NN graph into a large number of relatively small sub-clusters i.e., edge cut



K-NN Graph

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

Relative closeness: closeness of c_1 and c_2 over internal closeness

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

Relative interconnectivity: connectivity of c_1 and c_2 over internal connectivity

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

Probabilistic Hierarchical Clustering

- Algorithmic hierarchical clustering
  - 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: Mean and Variance

$$\mathcal{N}(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- The probability that a point $x_i \in X$ is generated by the model

$$P(x_i|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- 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 μ and σ^2 such that

the maximum likelihood

$$\mathcal{N}(\mu_0, \sigma_0^2) = \arg \max\{L(\mathcal{N}(\mu, \sigma^2) : X)\}$$

A Probabilistic Hierarchical Clustering Algorithm

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

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

  where *P*() is the maximum likelihood
- 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: $D = \{o_1, \ldots, o_n\}$: a data set containing n objects

  Output: A hierarchy of clusters

  Method

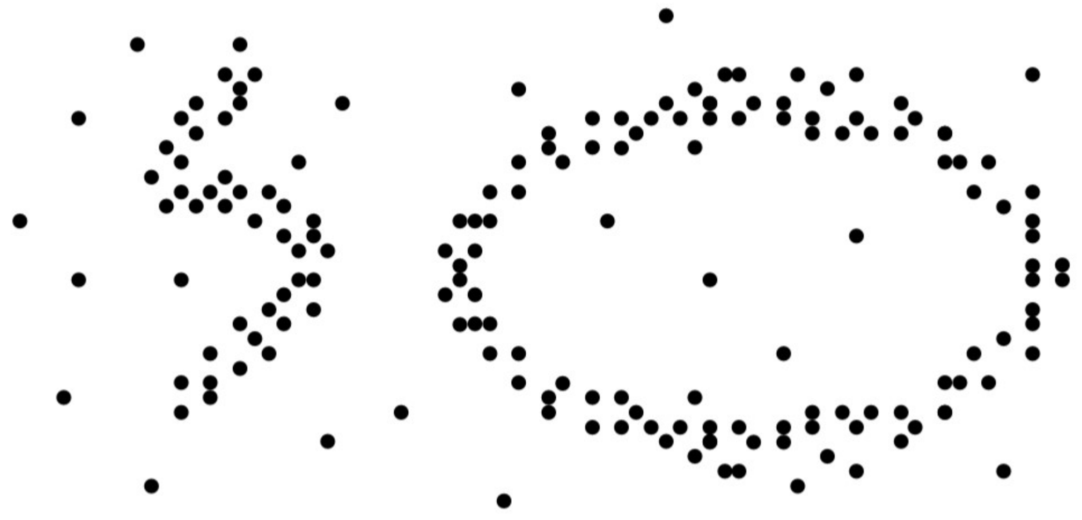
```
    Create a cluster for each object $C_i = \{o_i\}$, $1 \le i \le n$;
    For i = 1 to n {
        Find pair of clusters $C_i$ and $C_j$ such that
            $C_i, C_j = \text{argmax}_{i \neq j} \{\log (P(C_i \cup C_j)/(P(C_i)P(C_j)))\}$;
        If $\log (P(C_i \cup C_j)/(P(C_i)P(C_j))) > 0$ then merge $C_i$ and $C_j$ }
```

Module V - Cluster Analysis

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

Density-Based Clustering Methods

Partitioning and hierarchical methods are designed to find spherical-shaped clusters. They have difficulty finding clusters of arbitrary shape such as the "S" shape and oval clusters in Figure



Clusters of arbitrary shape.

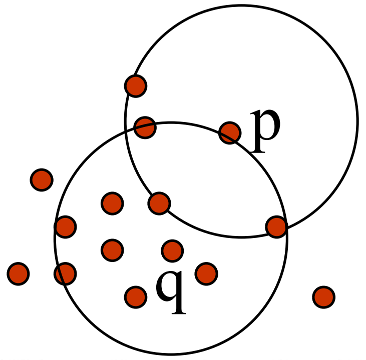
To find clusters of arbitrary shape, alternatively, we can model clusters as dense regions in the data space, separated by sparse regions. This is the main strategy behind *density-based clustering methods*, which can discover clusters of nonspherical shape.

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
- Basic techniques of Density-Based Clustering Methods are:
  - DBSCAN: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - DENCLUE: Hinneburg & D. Keim (KDD'98)

DBSCAN

- Density Based Spatial Clustering of Applications with Noise, which consists of two parameters*:*
  - *Eps(Epsilon)*: (ε - radius of the Circle)Maximum radius of the neighborhood
  - *MinPts*: Minimum number of points in an ε -neighborhood of that point



MinPts = 5

Eps = 1 cm

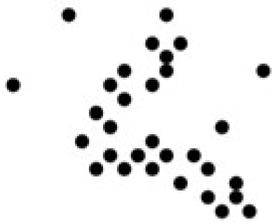
MinPts, which specifies the density threshold of dense regions.

Core-Object/Core Point

An object is a **core object** if the ϵ-neighborhood of the object contains at least *MinPts* objects.

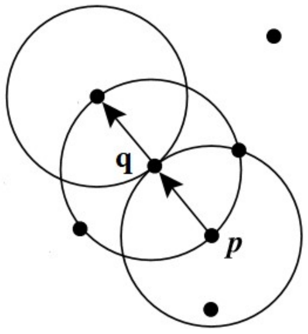
DBSCAN

Given a set, *D*, of objects, we can identify all core objects with respect to the given parameters, ϵ and *MinPts*.



Directly Density-reachable:

p is directly density-reachable from another object ***q*** if and only if ***q*** is a core object and ***p*** is in the ϵ-neighborhood of ***q***.

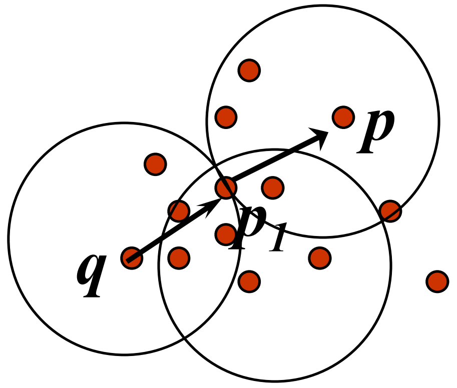


MinPts = 5

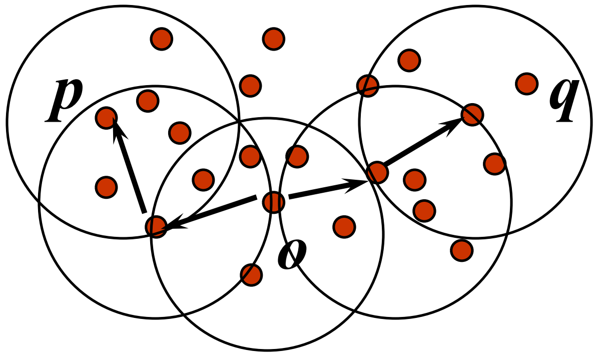
Eps = 1 cm

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

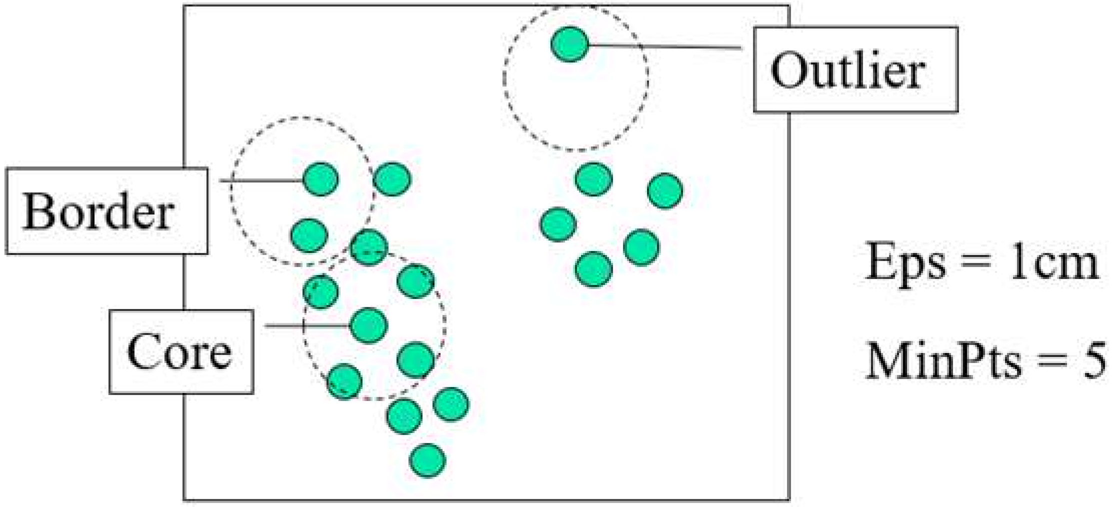


- **Density-connected:**
  - A point *p* is density-connected to a point *q* w.r.t. *Eps*, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. *Eps* and *MinPts*



DBSCAN

- **Core Point :** An object is a core point if the Eps-neighbourhood of the object contains at least MinPts objects.
- **Noise Point :** Outlier point
- **Border Point :**If *p* is a border point, no points are density-reachable from *p* and DBSCAN visits the next point of the database



"*How does DBSCAN find clusters?*" Initially, all objects in a given data set *D* are marked as "unvisited." DBSCAN randomly selects an unvisited object ***p***, marks ***p*** as "visited," and checks whether the ϵ-neighborhood of ***p*** contains at least *MinPts* objects. If not, ***p*** is marked as a noise point.

OPTICS

- OPTICS: Ordering Points To Identify the Clustering Structure
  - Ankerst, Breunig, Kriegel, and Sander (SIGMOD'99)
  - Produces a special order of the database w.r.t its density-based clustering structure.
  - DBSCAN fixes ε and *MinPts* where as OPTICS does not fix the ε but fix the *MinPts.*
  - OPTICS produces the output in the form of cluster ordering.
  - Cluster ordering uses density based clustering technique to maintains the linear list (higher to lower density) of all cluster objects.

OPTICS

- OPTICS needs two important pieces of information per object:
- Core-Distance:

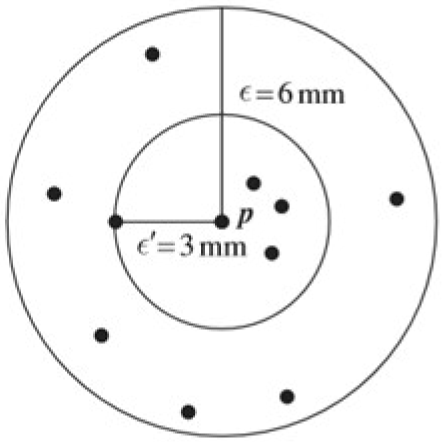
The **core-distance** of an object p is the smallest value ϵ' such that the ϵ'-neighborhood of p has at least *MinPts* objects. That is, ϵ' is the minimum distance threshold that makes p a core object. If p is not a core object with respect to ϵ and *MinPts*, the core-distance of p is undefined.

- Reachability-Distance:

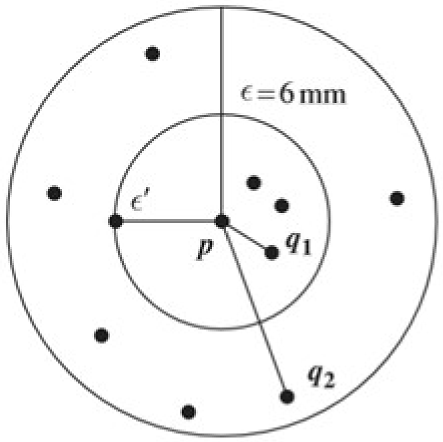
The **reachability-distance** to object p from q is the minimum radius value that makes p density-reachable from q. According to the definition of density-reachability, q has to be a core object and p must be in the neighborhood of q. Therefore, the reachability-distance from q to p is $\max\{\textit{core-distance}(q), \textit{dist}(p, q)\}$. If q is not a core object with respect to ϵ and *MinPts*, the reachability-distance to p from q is undefined.

OPTICS Example

Core-distance and reachability-distance. Figure illustrates the concepts of core-distance and reachability-distance. Suppose that $\epsilon = 6$ mm and $MinPts = 5$. The core-distance of $\boldsymbol{p}$ is the distance, ϵ', between $\boldsymbol{p}$ and the fourth closest data object from $\boldsymbol{p}$. The reachability-distance of $\boldsymbol{q_1}$ from $\boldsymbol{p}$ is the core-distance of $\boldsymbol{p}$ (i.e., $\epsilon' = 3$ mm) because this is greater than the Euclidean distance from $\boldsymbol{p}$ to $\boldsymbol{q_1}$. The reachability-distance of $\boldsymbol{q_2}$ with respect to $\boldsymbol{p}$ is the Euclidean distance from $\boldsymbol{p}$ to $\boldsymbol{q_2}$ because this is greater than the core-distance of $\boldsymbol{p}$.



Core-distance of p



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

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

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

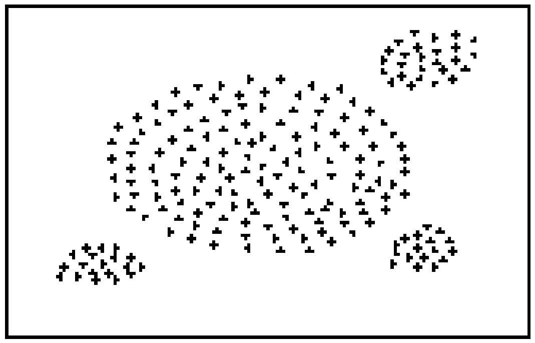
gradient of x in the direction of x_i

- 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

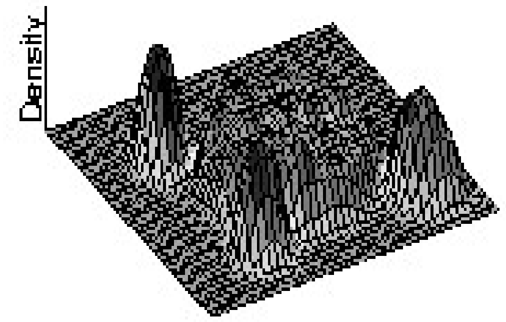
Denclue: Technical Essence

- **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

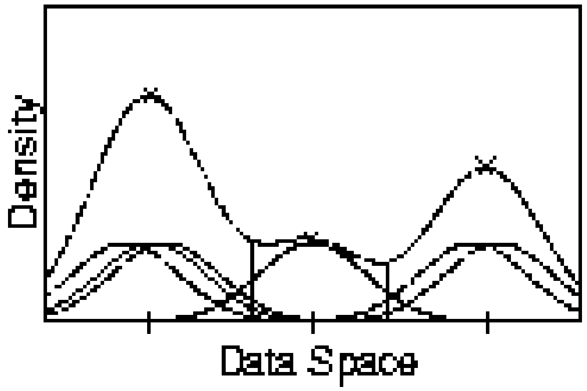
Density Attractor



(a) Data Set



(c) Gaussian



Module V - Cluster Analysis

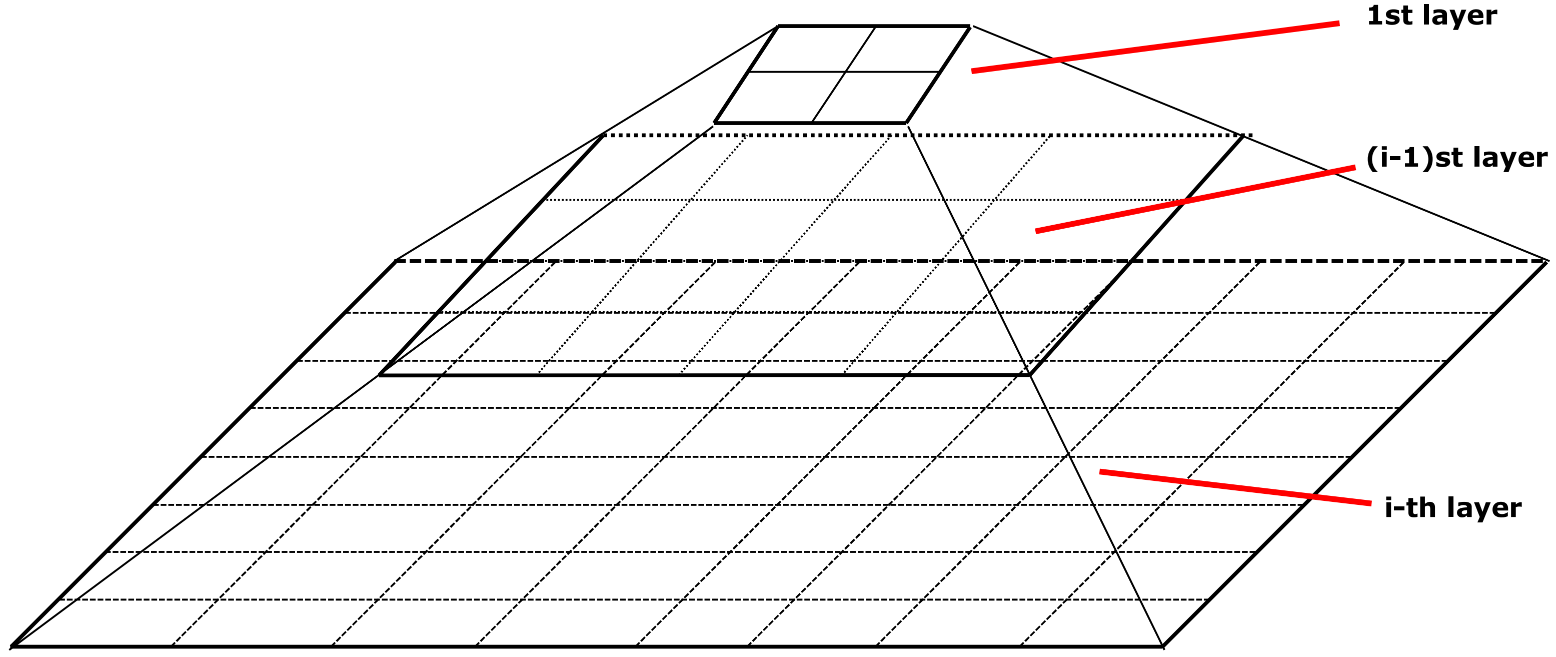
- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering

Grid-Based Clustering Method

- Using multi-resolution grid data structure
- Several interesting methods
  - STING (a STatistical INformation Grid approach) by Wang, Yang and Muntz (1997)
  - 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*, *stdev*, *min*, *max*
  - type of distribution—*normal*, *uniform*, exponential,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

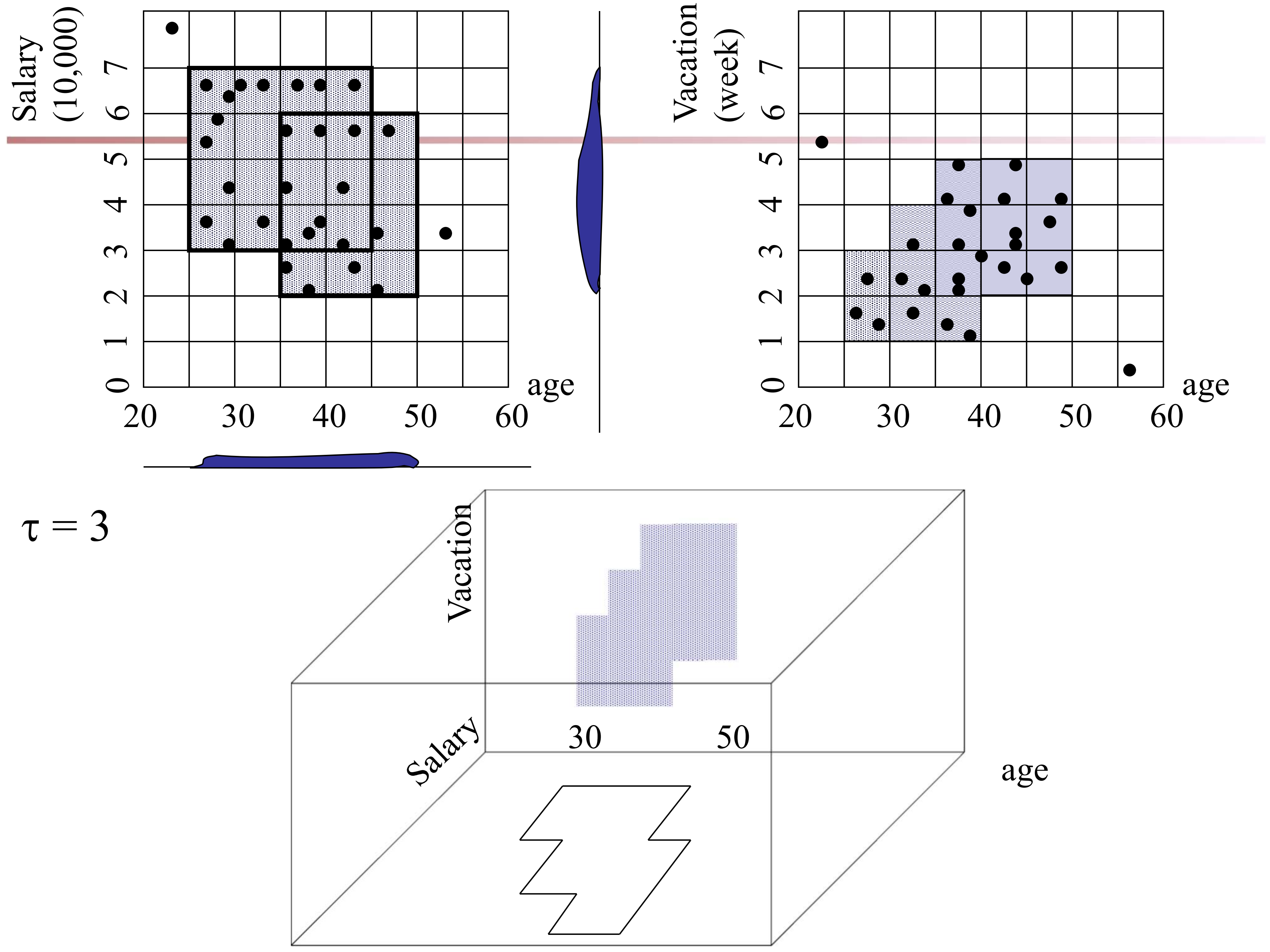
- Remove the irrelevant cells from further consideration
- When finish examining the current layer, proceed to the next lower level
- Repeat this process until the bottom layer is reached
- Advantages:
  - Query-independent, easy to parallelize, incremental update
  - *O(K),* where *K* is the number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



Salary (10,000)
0
1
2
3
4
5
6
7
20
30
40
50
60
age
Vacation (week)
0
1
2
3
4
5
6
7
20
30
40
50
60
age
Vacation
Salary
30
50
age

$\tau = 3$

Strength and Weakness of *CLIQUE*

- Strength
  - *automatically* finds subspaces of the highest dimensionality such that high density clusters exist in those subspaces
  - 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

Module V - Cluster Analysis

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

Assessing Clustering Tendency

- 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, $p_1, \ldots, p_n$, 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, $q_1, \ldots, q_n$, 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:

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

  - 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 highly skewed(inaccurate), H is close to 0

Determine the Number of Clusters

- Empirical method
  - # of clusters ≈√n/2 for a dataset of n points
- Elbow method

The **elbow method** is based on the observation that increasing the number of clusters can help to reduce the sum of within-cluster variance of each cluster.
Having more clusters allows one to capture finer groups of data objects that are more similar to each other.

- 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 and 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