


# **Cluster analysis:**

---

# **Concepts and Techniques**

# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

- Cluster Analysis: Basic Concepts 
- Partitioning Methods
- Hierarchical Methods
- Density based Methods
- Introduction to grid based method
- 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**
  - It is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other and than to those in other groups(clusters)
  - Known as
    - Segmentation- (as it partitioning large datasets in to groups according to their similaritis)
- **Unsupervised learning:** clustering do not rely on predefin classes and class labeled training examples.
- **Typical applications**
  - As a **stand-alone tool** to get insight into data distribution
  - As a **preprocessing step** for other algorithms

# Clustering: Application Examples

---

- 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
- Climate: understanding earth climate, find patterns of atmospheric and ocean

# Requirements for cluster analysis 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
  - Requirements for domain knowledge to determine input parameters
  - 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) vs. connectivity-based (e.g., density)
- 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
  - 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

# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

- Cluster Analysis: Basic Concepts
- Partitioning Methods 
- Hierarchical Methods
- Density based Methods
- Introduction to grid based method
- Summary



# Partitioning Algorithms: Basic Concept

---

- Partitioning method: Partitioning a database  $D$  of  $n$  objects into a set of  $k$  clusters based on a distance function.
- Global optimal: exhaustively enumerate all partitions
- Heuristic methods: *k-means* and *k-medoids* algorithms

# The *K-Means* Clustering Method

---

- The k-means algorithm partitions the given data into k clusters.
  - Each cluster has a cluster center, called cluster **centroid**.
  - The centroid, usually used to represent the cluster is simply the mean of all the data points in the cluster, which gives the name to the algorithm e.g k-means algorithm
  - k is specified by the user

# K-means algorithm

---

- Given  $k$ , the k-means algorithm works as follows:
  - 1) Randomly choose  $k$  data points (seeds) to be the initial centroids, cluster centers
  - 2) Assign each data point to the closest centroid
  - 3) Re-compute the centroids using the current cluster memberships.
  - 4) If a stopping criterion is not met, go to 2).

# K-means algorithm – (cont ...)

---

**Algorithm**  $k$ -means( $k, D$ )

- 1 Choose  $k$  data points as the initial centroids (cluster centers)
- 2 **repeat**
- 3     **for** each data point  $\mathbf{x} \in D$  **do**
- 4         compute the distance from  $\mathbf{x}$  to each centroid;
- 5         assign  $\mathbf{x}$  to the closest centroid         // a centroid represents a cluster
- 6     **endfor**
- 7     re-compute the centroids using the current cluster memberships
- 8 **until** the stopping criterion is met

# Stopping/convergence criterion

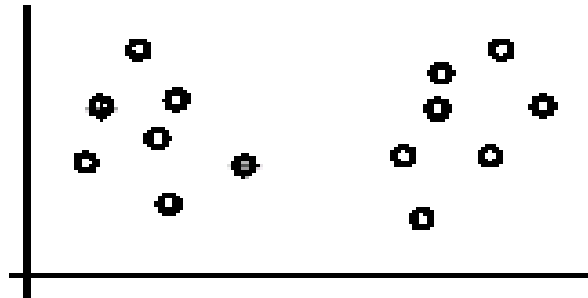
---

1. no (or minimum) re-assignments of data points to different clusters,
2. no (or minimum) change of centroids, or
3. minimum decrease in the **sum of squared error** (SSE),

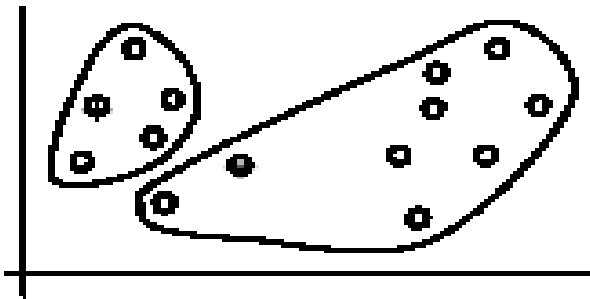
$$SSE = \sum_{j=1}^k \sum_{\mathbf{x} \in C_j} dist(\mathbf{x}, \mathbf{m}_j)^2 \quad (1)$$

1.  $C_j$  is the  $j$ th cluster,  $\mathbf{m}_j$  is the centroid of cluster  $C_j$  (the mean vector of all the data points in  $C_j$ ), and  $dist(\mathbf{x}, \mathbf{m}_j)$  is the distance between data point  $\mathbf{x}$  and centroid  $\mathbf{m}_j$ .

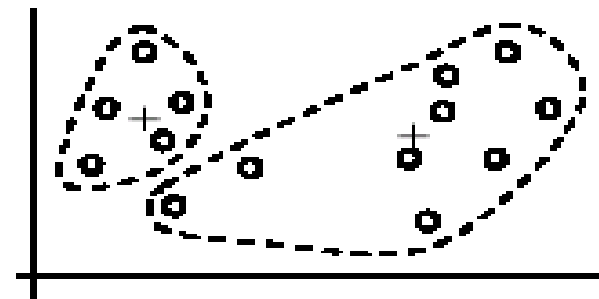
# An example



(A). Random selection of  $k$  centers

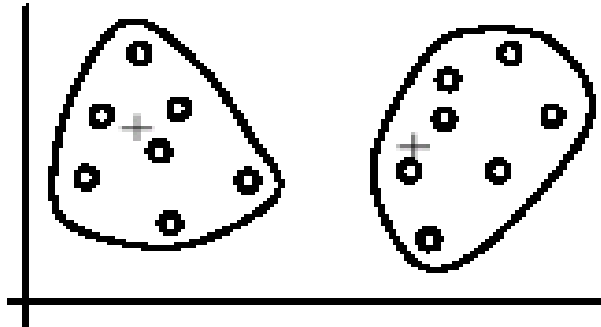


Iteration 1: (B). Cluster assignment

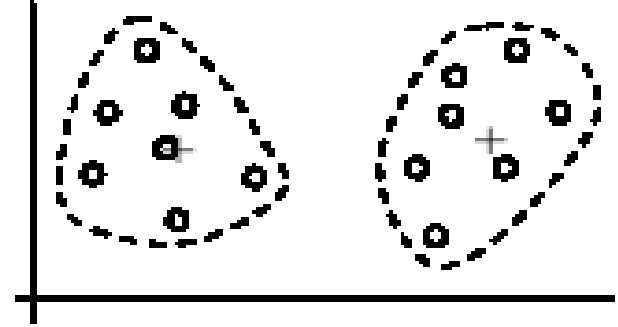


(C). Re-compute centroids

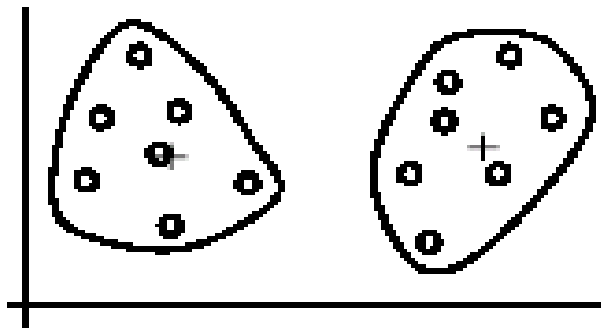
# An example (cont ...)



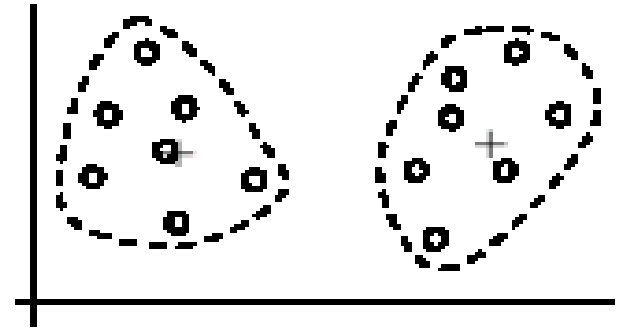
Iteration 2: (D). Cluster assignment



(E). Re-compute centroids



Iteration 3: (F). Cluster assignment



(G). Re-compute centroids

# An example distance function

---

The  $k$ -means algorithm can be used for any application data set where the **mean** can be defined and computed. In the **Euclidean space**, the mean of a cluster is computed with:

$$\mathbf{m}_j = \frac{1}{|C_j|} \sum_{\mathbf{x}_i \in C_j} \mathbf{x}_i \quad (2)$$

where  $|C_j|$  is the number of data points in cluster  $C_j$ . The distance from one data point  $\mathbf{x}_i$  to a mean (centroid)  $\mathbf{m}_j$  is computed with

$$\begin{aligned} dist(\mathbf{x}_i, \mathbf{m}_j) &= \| \mathbf{x}_i - \mathbf{m}_j \| \\ &= \sqrt{(x_{i1} - m_{j1})^2 + (x_{i2} - m_{j2})^2 + \dots + (x_{ir} - m_{jr})^2} \end{aligned} \quad (3)$$



# Strengths of k-means

---

- Strengths:
  - Simple: easy to understand and to implement
  - Efficient: Time complexity:  $O(tkn)$ , where  $n$  is the number of data points,  $k$  is the number of clusters, and  $t$  is the number of iterations.
  - Since both  $k$  and  $t$  are small.  $k$ -means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.
- Note that: it terminates at a **local optimum** if SSE is used. The **global optimum** is hard to find due to complexity.

# Weaknesses of k-means

---

- The algorithm is only applicable if the **mean** is defined.
  - For categorical data, *k*-mode - the centroid is represented by most frequent values.
  - Handling categorical data: k-modes (Huang'98)
    - 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
- The user needs to specify *k*.
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values

# K-Medoids Method

---

- ▶ Minimize the sensitivity of k-means to outliers
- ▶ Pick actual objects to represent clusters instead of mean values
- ▶ Each remaining object is clustered with the representative object (**Medoid**) to which is the most similar
- ▶ The algorithm minimizes the sum of the dissimilarities between each object and its corresponding reference point

$$E = \sum_{i=1}^k \sum_{p \in C_i} |p - o_i|$$

- **E**: the sum of absolute error for all objects in the data set
- **P**: the data point in the space representing an object
- **O<sub>i</sub>**: is the representative object of cluster C<sub>i</sub>

# K-Medoids Method

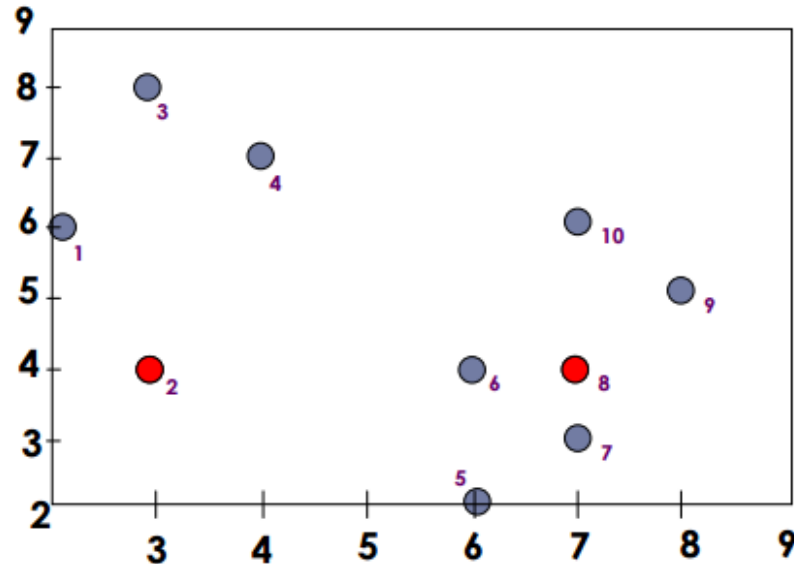
---

- Initial representatives are chosen randomly
- The iterative process of replacing representative objects by non representative objects continues as long as the quality of the clustering is improved
- For each representative Object O
  - For each non-representative object R, swap O and R
- Choose the configuration with the lowest cost
- Cost function is the difference in absolute error-value if a current representative object is replaced by a non-representative object

# K-Medoids Method: Example

## Data Objects

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



**Goal: create two clusters**

Choose randomly two medoids

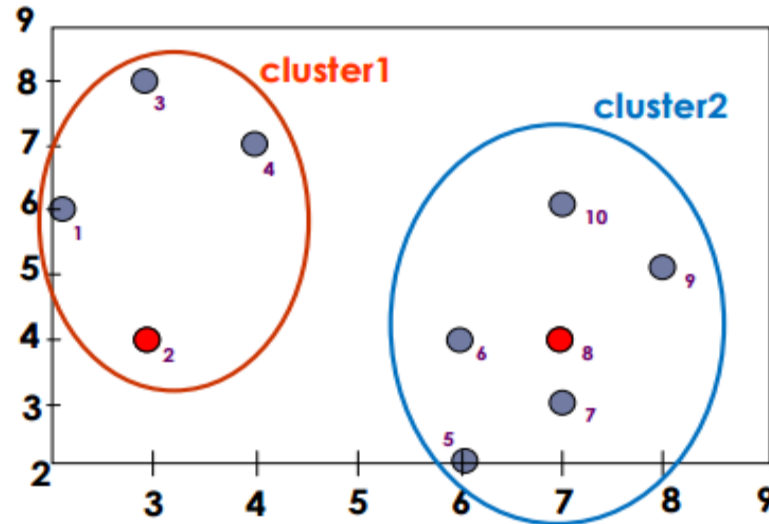
$$O_2 = (3, 4)$$

$$O_8 = (7, 4)$$

# K-Medoids Method: Example

## Data Objects

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



→ Assign each object to the closest representative object

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

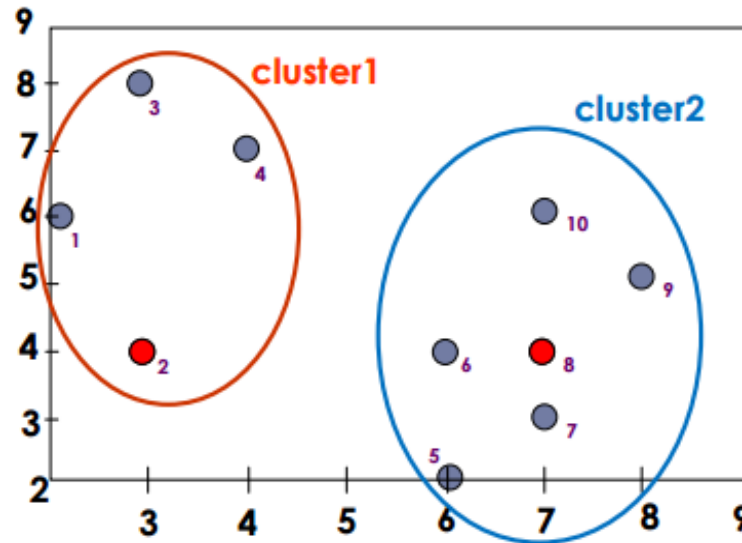
$$\text{Cluster1} = \{O_1, O_2, O_3, O_4\}$$

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

# K-Medoids Method: Example

Data Objects

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



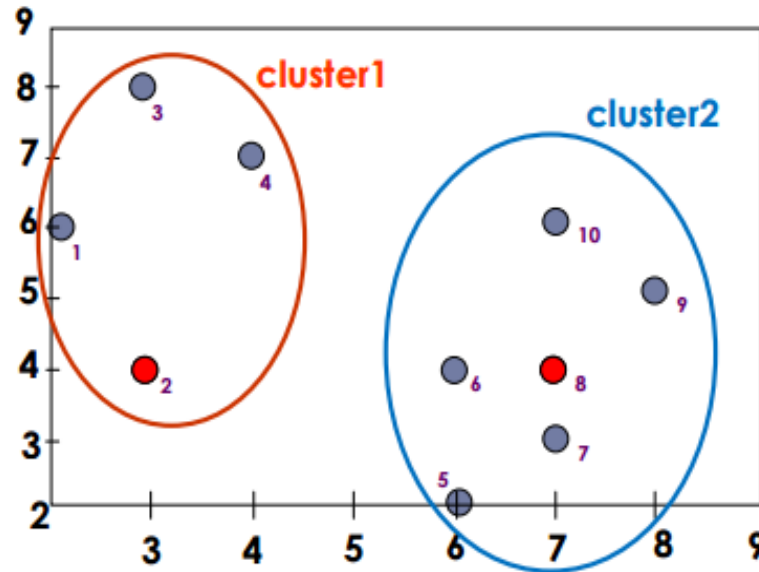
→ Compute the absolute error criterion [for the set of Medoids (O2,O8)]

$$E = \sum_{i=1}^k \sum_{p \in C_i} |p - o_i| = |o_1 - o_2| + |o_3 - o_2| + |o_4 - o_2| \\ + |o_5 - o_8| + |o_6 - o_8| + |o_7 - o_8| + |o_9 - o_8| + |o_{10} - o_8|$$

# K-Medoids Method: Example

## Data Objects

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



→ The absolute error criterion [for the set of Medoids ( $O_2, O_8$ )]

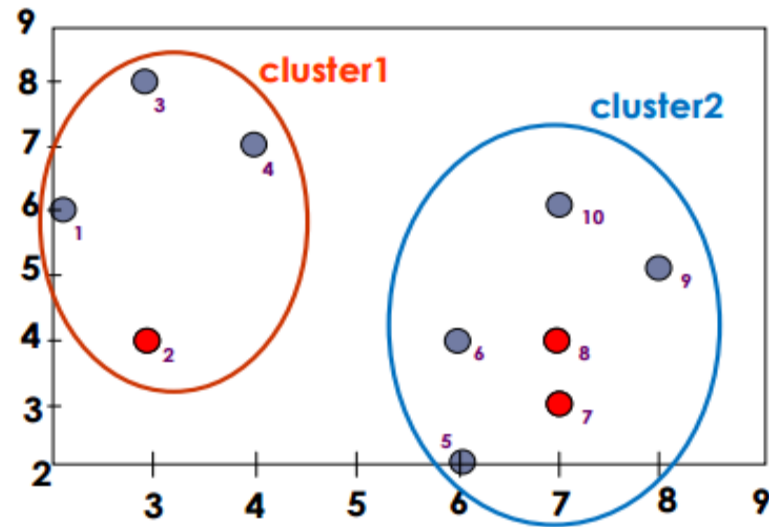
$$E = (3 + 4 + 4) + (3 + 1 + 1 + 2 + 2) = 20$$



# K-Medoids Method: Example

## Data Objects

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



→ Choose a random object  $O_7$

→ Swap  $O_8$  and  $O_7$

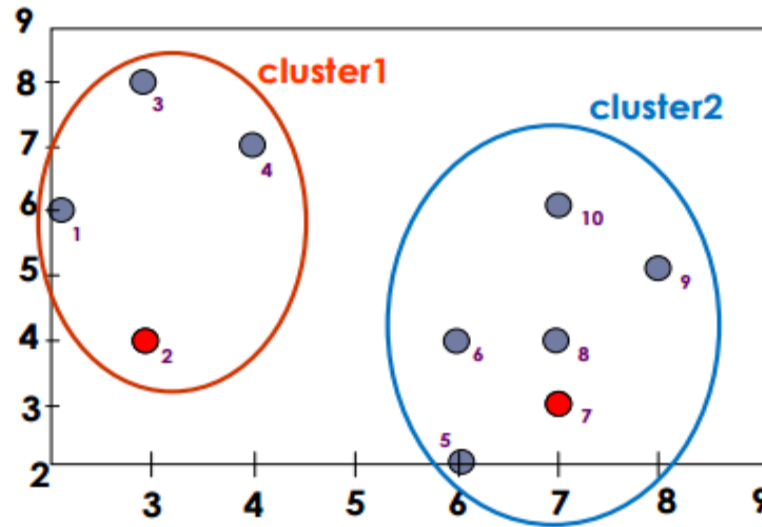
→ Compute the absolute error criterion [for the set of Medoids ( $O_2, O_7$ )]

$$E = (3 + 4 + 4) + (2 + 2 + 1 + 3 + 3) = 22$$

# K-Medoids Method: Example

## Data Objects

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



→ Compute the cost function

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

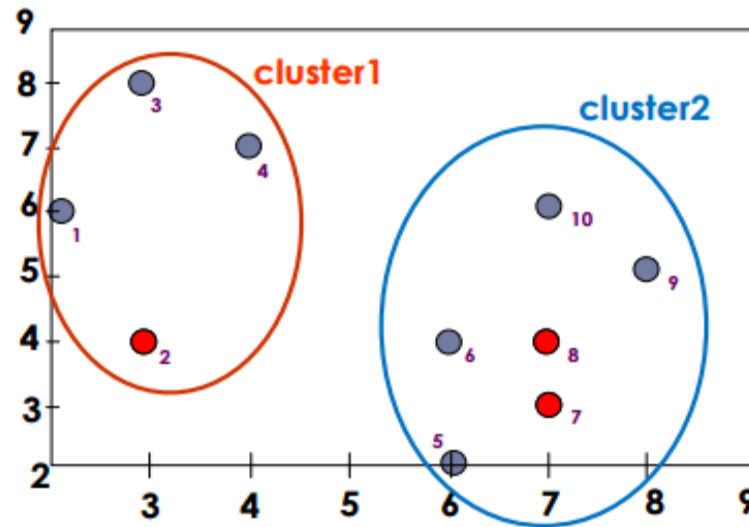
$$S = 22 - 20$$

$S > 0 \Rightarrow$  it is a bad idea to replace  $O_8$  by  $O_7$

# K-Medoids Method

## Data Objects

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



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

# K-Medoids Algorithm(PAM)

## PAM : Partitioning Around Medoids

### ► Input

- K: the number of clusters
- D: a data set containing n objects

### ► Output: A set of k clusters

### ► Method:

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

# K-Medoids Properties(k-medoids vs.K-means)

---

- ▶ The complexity of each iteration is  $O(k(n-k)^2)$
- ▶ For large values of  $n$  and  $k$ , such computation becomes very costly
- ▶ **Advantages**
  - K-Medoids method is more robust than k-Means in the presence of noise and outliers
- ▶ **Disadvantages**
  - K-Medoids is more costly than the k-Means method
  - Like k-means, k-medoids requires the user to specify  $k$
  - It does not scale well for large data sets

# K-means vs k-medoids.

---

- What is the difference between K-means and k-medoids?
- K-means attempts to minimize the total squared error, while k-medoids minimizes the sum of dissimilarities between points labeled to be in a cluster and a point designated as the center of that cluster. In contrast to the k -means algorithm, k -medoids chooses data points as centers ( medoids or exemplars).

# Algorithm: k-medoids.

## K-MEDOID EXAMPLE

K=2

i	x	y
x <sub>1</sub>	2	6
x <sub>2</sub>	3	4
x <sub>3</sub>	3	8
x <sub>4</sub>	4	7
x <sub>5</sub>	6	2
x <sub>6</sub>	6	4
x <sub>7</sub>	7	3
x <sub>8</sub>	7	4
x <sub>9</sub>	8	5
x <sub>10</sub>	7	6

### Step 1

We select two random representative objects:

$c_1(3, 4)$ ,  $c_2(7, 4)$

i	x	y	c <sub>1</sub>	Distance/cost	c
x <sub>1</sub>	2	6	3	$ 2-3  +  6-4 $	3
x <sub>3</sub>	3	8	3	$0 + 4$	4
x <sub>4</sub>	4	7	3	$1 + 3$	4
x <sub>5</sub>	6	2	3	$3 + 2$	5
x <sub>6</sub>	6	4	3	$3 + 0$	3
x <sub>7</sub>	7	3	3	$4 + 1$	5
x <sub>9</sub>	8	5	3	$5 + 1$	6
x <sub>10</sub>	7	6	3	$4 + 2$	6

$m = (a, b)$

$n = (c, d)$

Distance =  $|a - c| + |b - d|$

i	x	y	c <sub>2</sub>	Distance/cost	c
x <sub>1</sub>	2	6	7	$ 2-7  +  6-4 $	7
x <sub>3</sub>	3	8	7	$4 + 4$	8
x <sub>4</sub>	4	7	7	$3 + 3$	6
x <sub>5</sub>	6	2	7	$1 + 2$	3
x <sub>6</sub>	6	4	7	$1 + 0$	1
x <sub>7</sub>	7	3	7	$0 + 1$	1
x <sub>9</sub>	8	5	7	$1 + 1$	2
x <sub>10</sub>	7	6	7	$0 + 2$	2

Compare cost of  $\text{Cost}(c_1)$  and  $\text{Cost}(c_2)$  for every i & Select the minimum one

# Algorithm: k-medoids.

## K-MEDOID EXAMPLE

K=2

i	x	y
x <sub>1</sub>	2	6
x <sub>2</sub>	3	4
x <sub>3</sub>	3	8
x <sub>4</sub>	4	7
x <sub>5</sub>	6	2
x <sub>6</sub>	6	4
x <sub>7</sub>	7	3
x <sub>8</sub>	7	4
x <sub>9</sub>	8	5
x <sub>10</sub>	7	6

### Step 1

We select two random representative objects:

$$c_1(3,4), \quad c_2(7,4)$$

Step II) then cluster are

cluster 1:  $\{(2,6), (3,8), (4,7), (3,4)\}$

cluster 2:  $\{(7,4), (6,2), (6,4), (7,3), (8,5), (7,6)\}$

Calculate total cost

$$T \text{ cost}(x, c) = \sum_{i=1}^d |x_i - c_i|$$

$$\begin{aligned} \text{Total cost} &= \{ \text{cost}((3,4), (2,6)), \text{cost}((3,4), (3,8)), \\ &\quad \text{cost}((3,4), (4,7)), \text{cost}((7,4), (8,5)), \\ &\quad \text{cost}((7,4), (6,2)), \text{cost}((7,4), (6,4)), \\ &\quad \text{cost}((7,4), (7,3)), \text{cost}((7,4), (7,6)) \} \\ &= (3+4+4) + (3+1+1+2+2) \\ &= 20 \end{aligned}$$

Step 3) Select one of non-medoids O'

Let's  $O' = (7,3)$  i.e.  $x_7$



# Algorithm: k-medoids.

So now medoid's are  $C(3,4)$  &  $O'(7,3)$

$i$	$x$	$y$	$O'$	Distance/cost	$C$
$x_1$	2	6	7 3	$(2-7) +  6-3 $	8
$x_3$	3	8	7 3	$4+5$	9
$x_4$	4	7	7 3	$3+4$	7
$x_5$	6	2	7 3	$1+1$	2
$x_6$	6	4	7 3	$1+1$	2
$x_8$	7	4	7 3	$0+1$	1
$x_9$	8	5	7 3	$1+2$	3
$x_{10}$	7	6	7 3	$0+3$	3

Compare the cost of cost( $C_i$ ) and cost( $O'$ ) for every  $i$  & Select the minimum one

$i$	$x$	$y$	$C_i$	Distance/cost	$C$
$x_1$	2	6	3 4	$ 2-3  +  6-4 $	= 3
$x_3$	3	8	3 4	$0+4$	= 4
$x_4$	4	7	3 4	$1+3$	= 4
$x_5$	6	2	3 4	$3+2$	= 5
$x_6$	6	4	3 4	$3+0$	= 3
$x_8$	7	4	3 4	$4+0$	= 4
$x_9$	8	5	3 4	$5+1$	= 6
$x_{10}$	7	6	3 4	$4+2$	= 6

Again create the cluster

cluster 1:  $\{(3,4), (2,6), (3,8), (4,7)\}$

cluster 2:  $\{(7,3), (6,2), (5,4), (7,4), (8,5), (7,6)\}$

$$\begin{aligned} \text{current total cost} &= (3+4+4) + (2+2+1+3+3) \\ &= 11 + 11 \\ &= 22 \end{aligned}$$


**Step 4)** So cost of swapping medoid from  $C_2$  to  $O'$  is

$$\begin{aligned} S &= \text{current total cost} - \text{past total cost} \\ &= 22 - 20 \\ &= 2 > 0 \end{aligned}$$

So, moving  $O'$  would be a bad idea so previous choice was Good

# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods 
- Density based Methods
- Introduction to grid based method
- Summary

# Hierarchical Clustering

## HIERARCHICAL CLUSTERING:

groups the data into tree of clusters

↓  
dendrogram  
↓

has sequences of all merges & splits

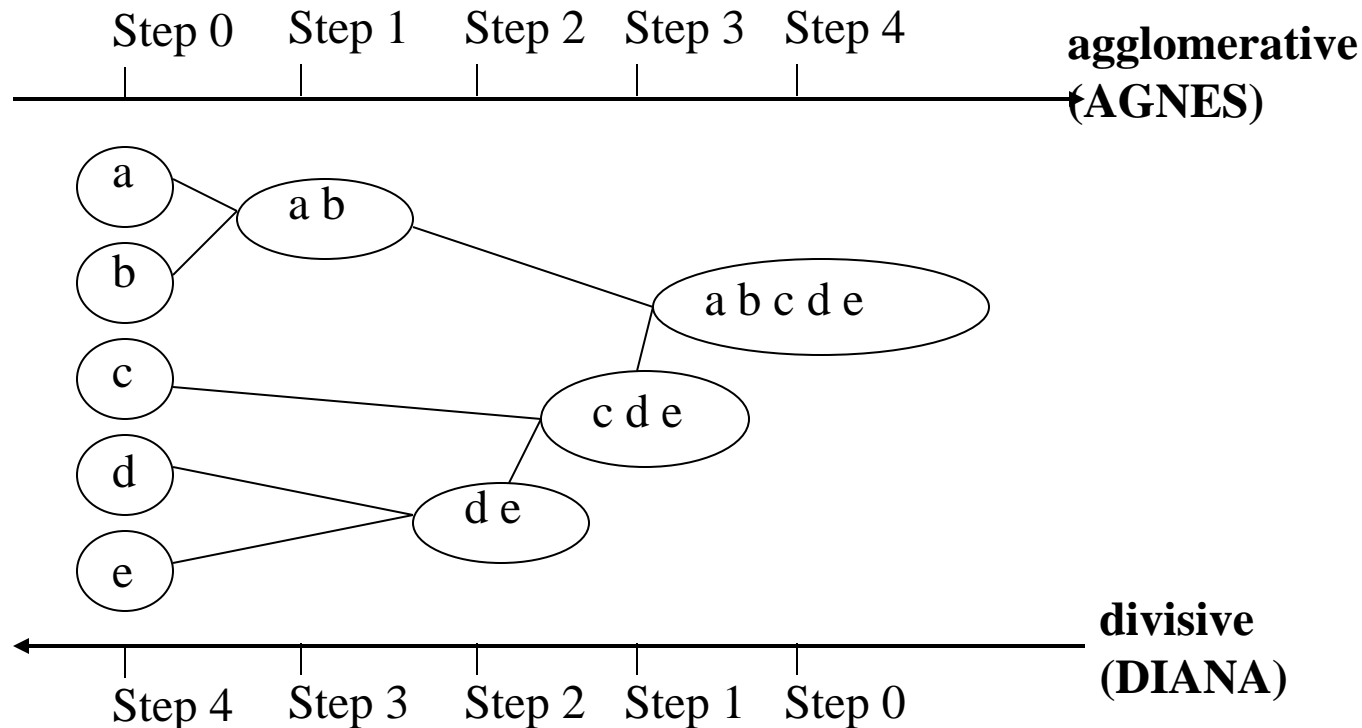
Leads { G.P.  
na { P.  
Team { C  
✓

### 2 methods

1. Agglomerative method
2. Divisive method

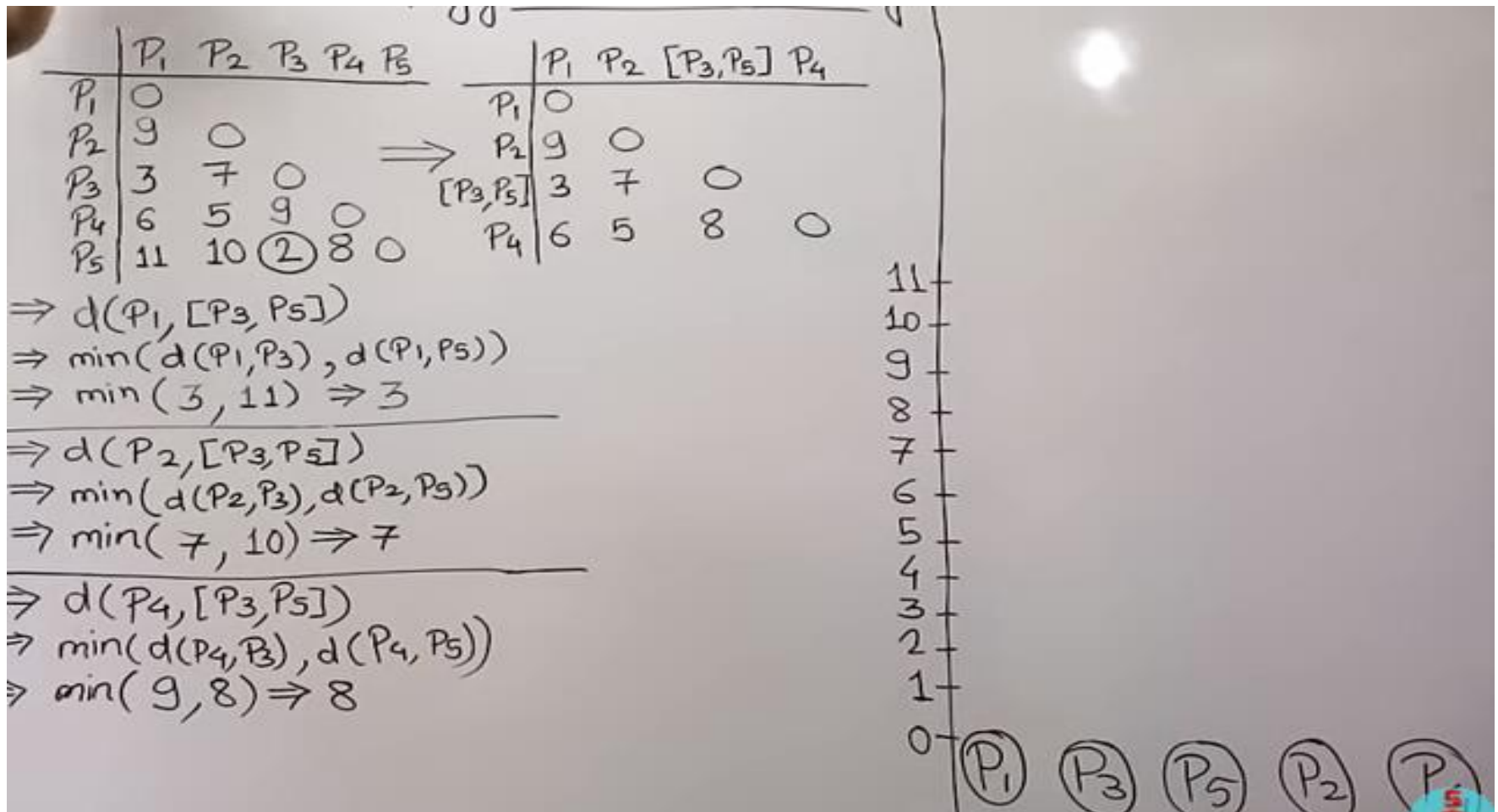
# Hierarchical Clustering

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



# Hierarchical Clustering

## ■ Agglomerative (Single Linkage)



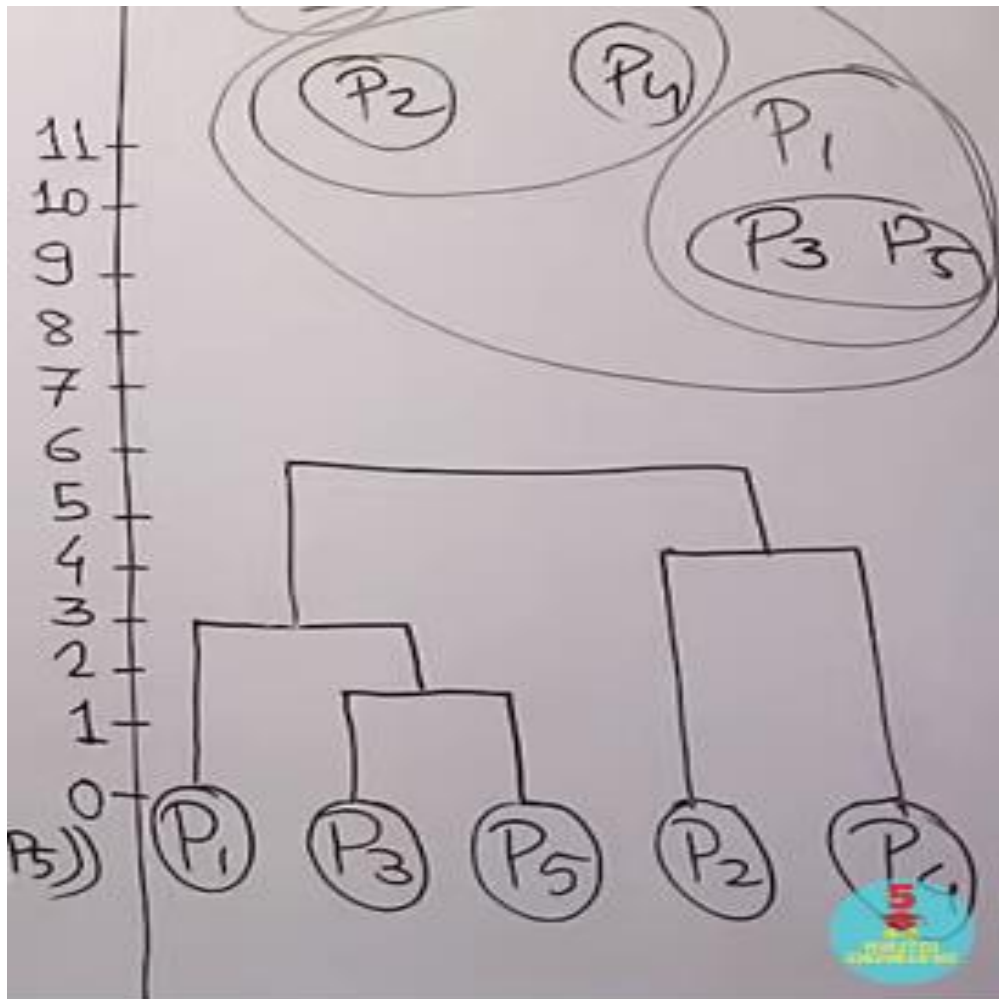
- **Agglomerative (Single Linkage)**





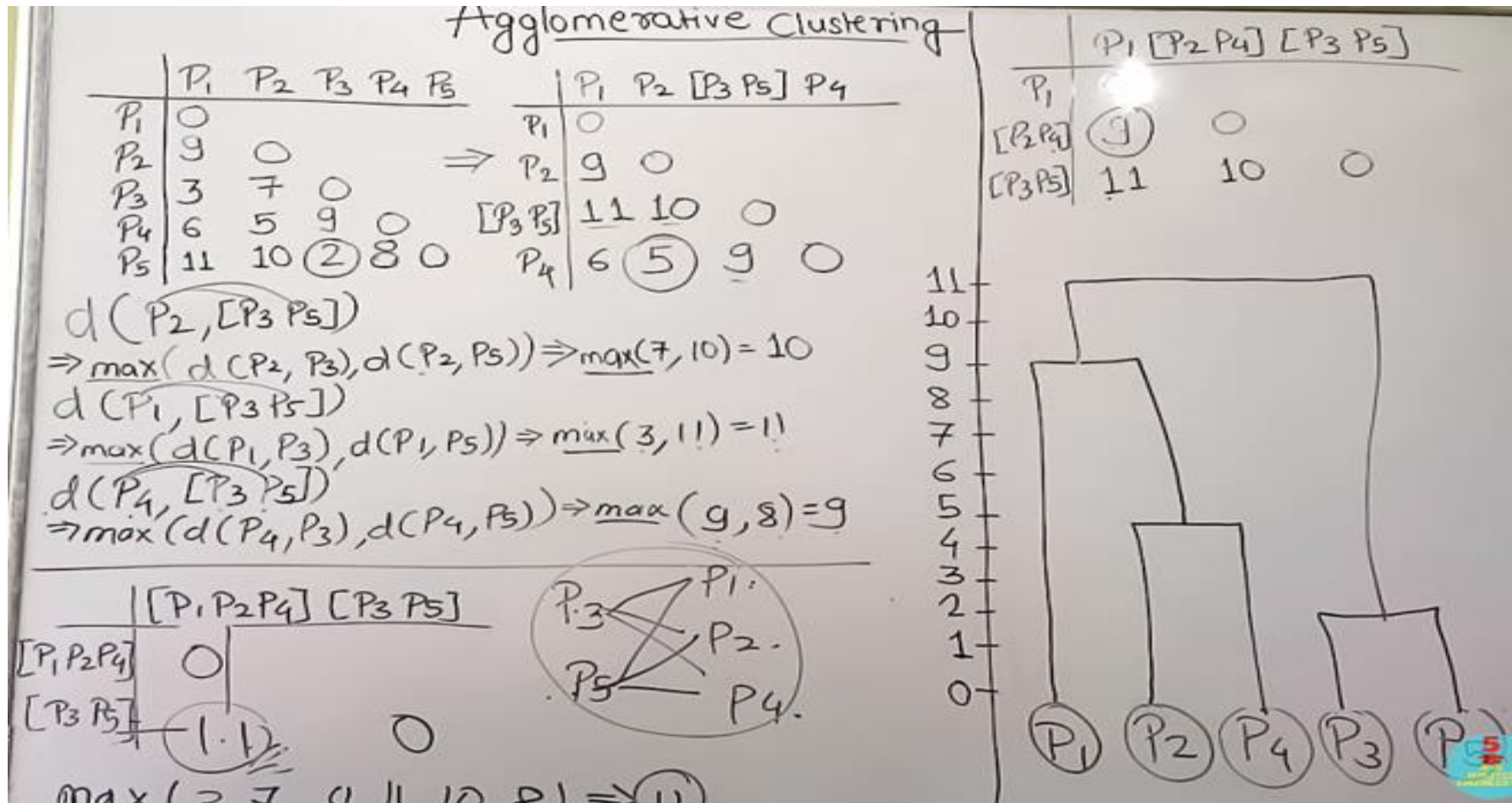
# Hierarchical Clustering

- Agglomerative (Single Linkage)



# Hierarchical Clustering

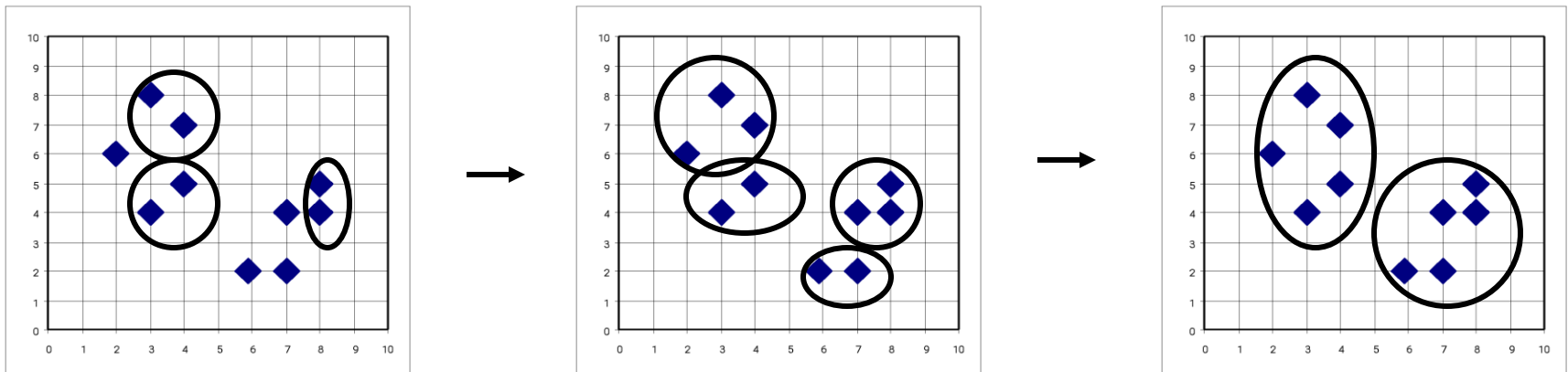
## ■ Agglomerative (Complete Linkage)



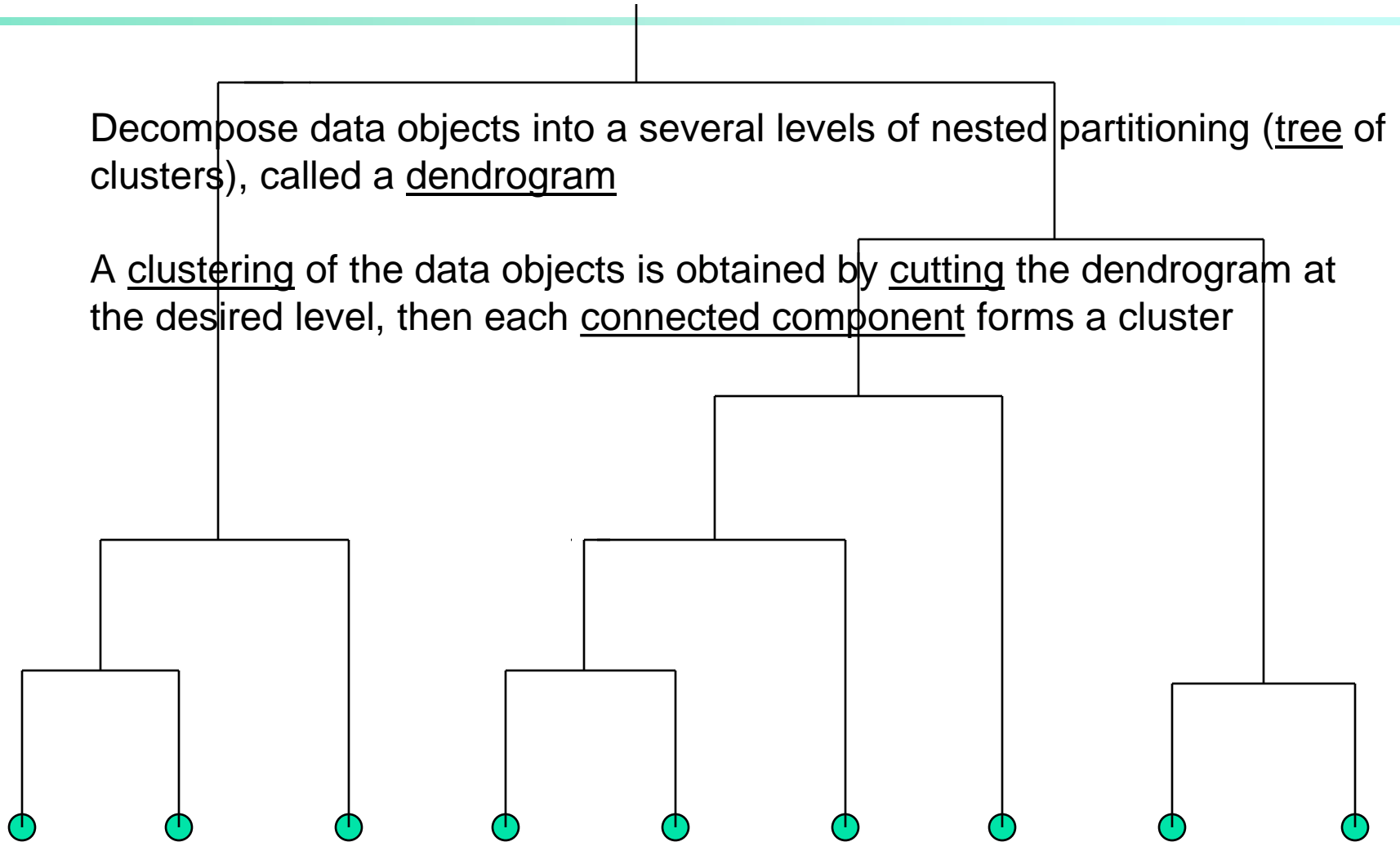


# AGNES (Agglomerative Nesting)

- 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

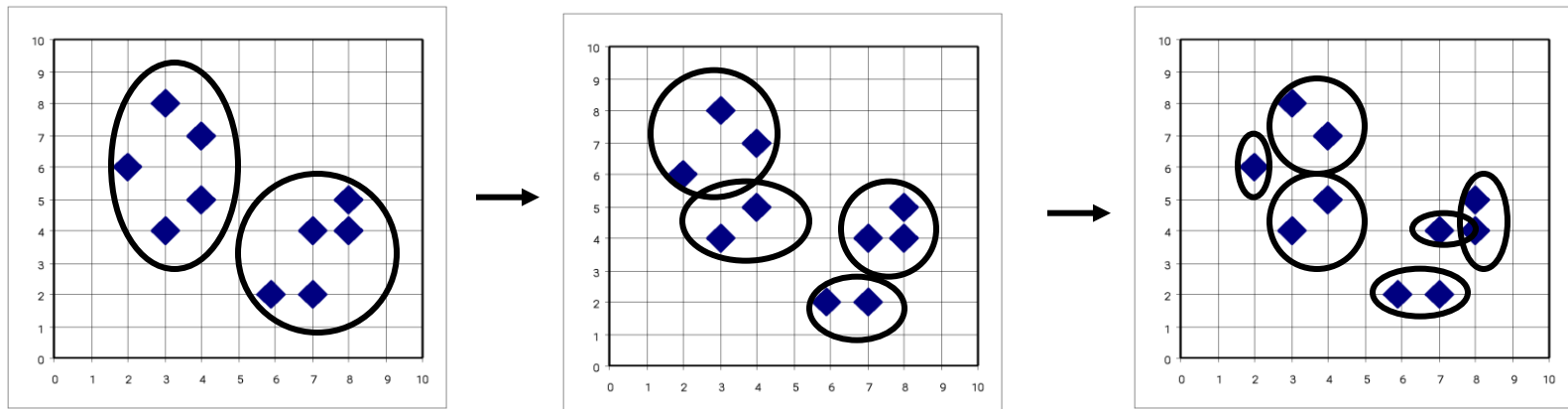


# Dendrogram: Shows How Clusters are Merged



# 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

---

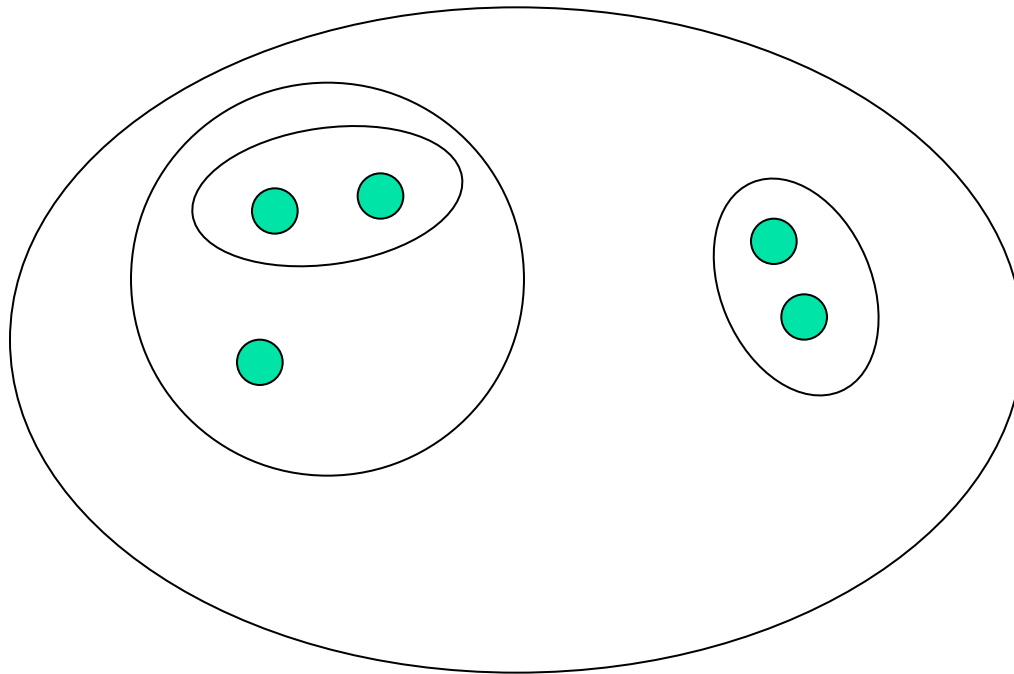
- Major weakness of agglomerative clustering methods
  - Can never undo what was done previously
  - Do not scale well: time complexity of at least  $O(n^2)$ , where  $n$  is the number of total objects
- Integration of hierarchical & distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally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

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

# BIRCH concepts and terminology

---

## Hierarchical clustering



# BIRCH concepts and terminology

---

## Hierarchical clustering

- The algorithm starts with single point clusters (every point in a database is a cluster).
  - Then it groups the closest points into separate clusters, and continues, until only one cluster remains.
-

# BIRCH concepts and terminology

---

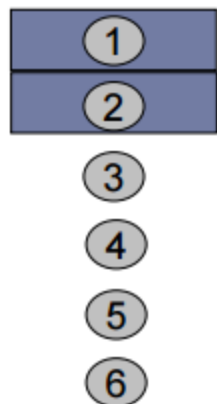
## Clustering Feature

- The BIRCH algorithm builds a clustering feature tree (CF tree) while scanning the data set.
  - Each entry in the CF tree represents a cluster of objects and is characterized by a triple (N, LS, SS).
-

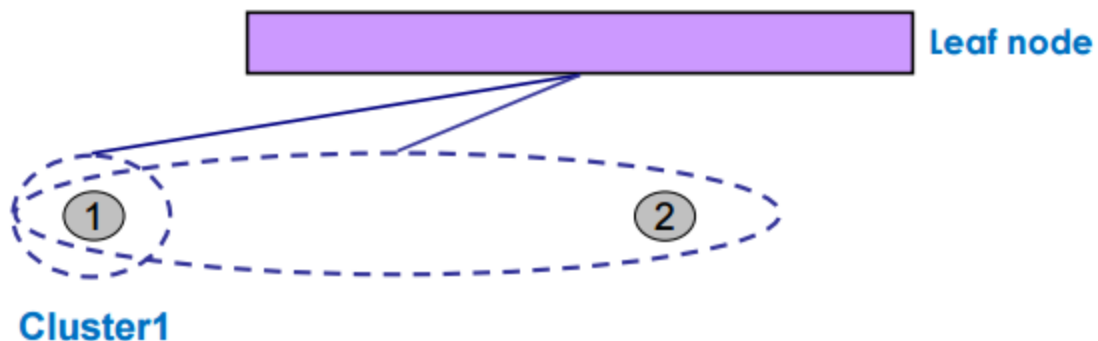


## BIRCH: The Idea by example

Data Objects



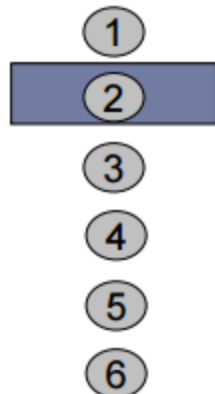
Clustering Process (build a tree)



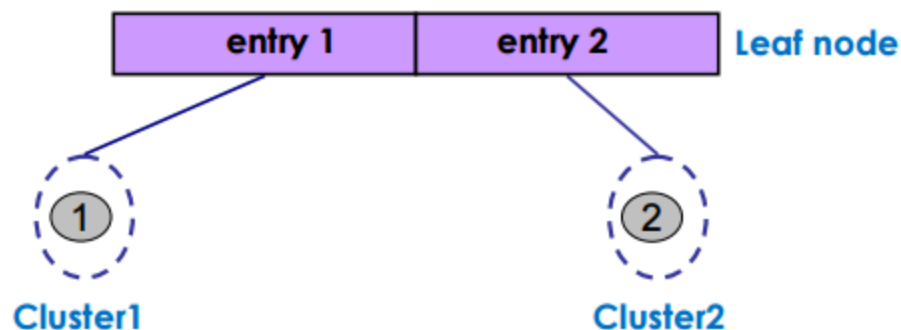
If cluster 1 becomes too large (not compact) by adding object 2, then split the cluster

## BIRCH: The Idea by example

Data Objects



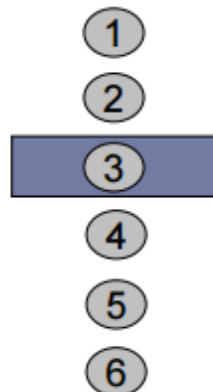
Clustering Process (build a tree)



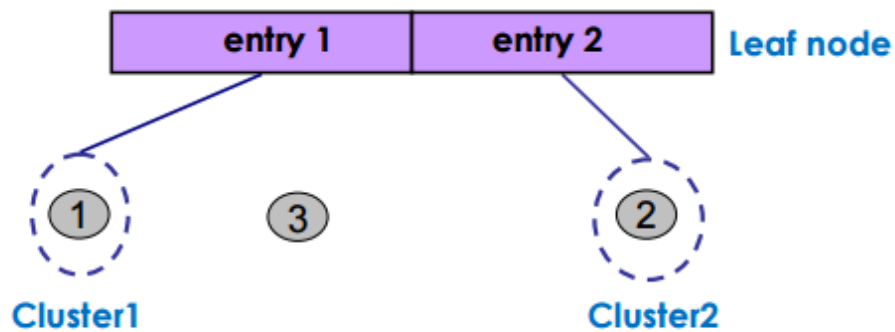
Leaf node with two entries

## BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)

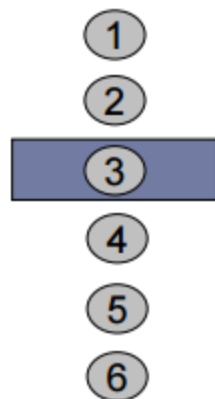


entry1 is the closest to object 3

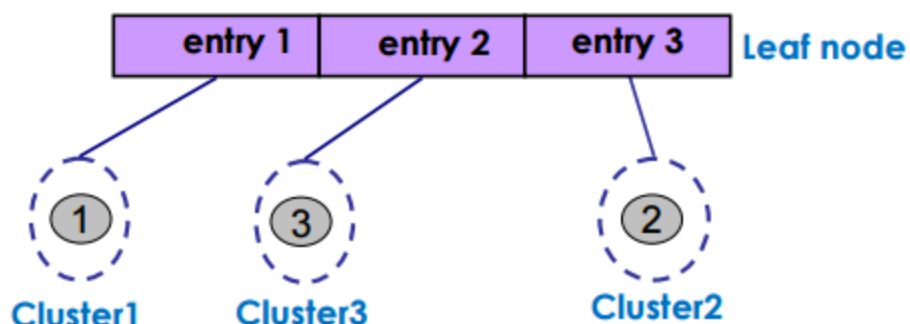
If cluster 1 becomes too large by adding object 3,  
then split the cluster

## BIRCH: The Idea by example

Data Objects



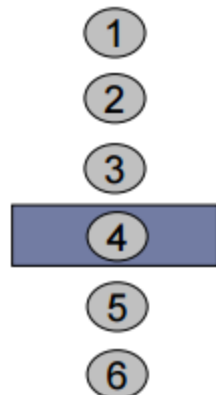
Clustering Process (build a tree)



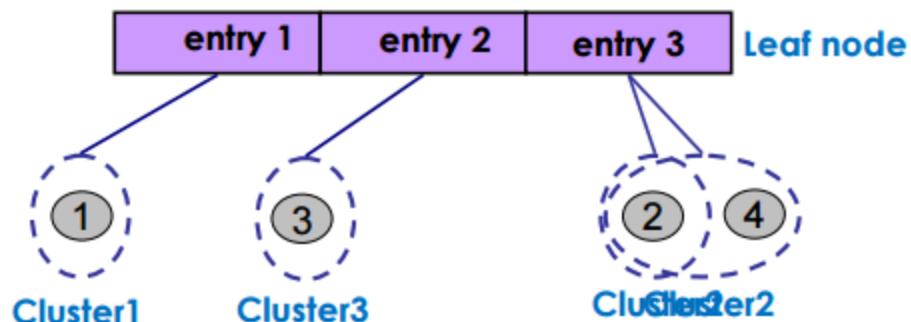
Leaf node with three entries

## BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)

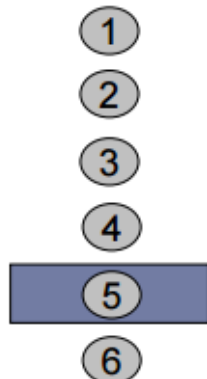


entry3 is the closest to object 4

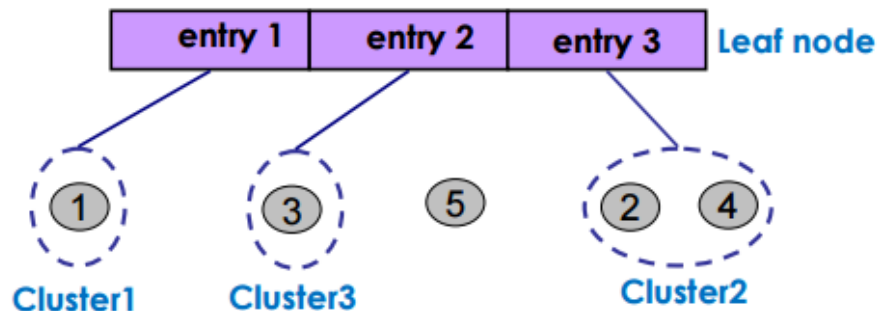
Cluster 2 remains compact when adding object 4  
then add object 4 to cluster 2

## BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)



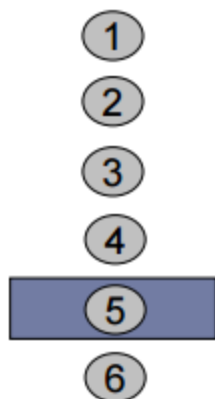
entry2 is the closest to object 5

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

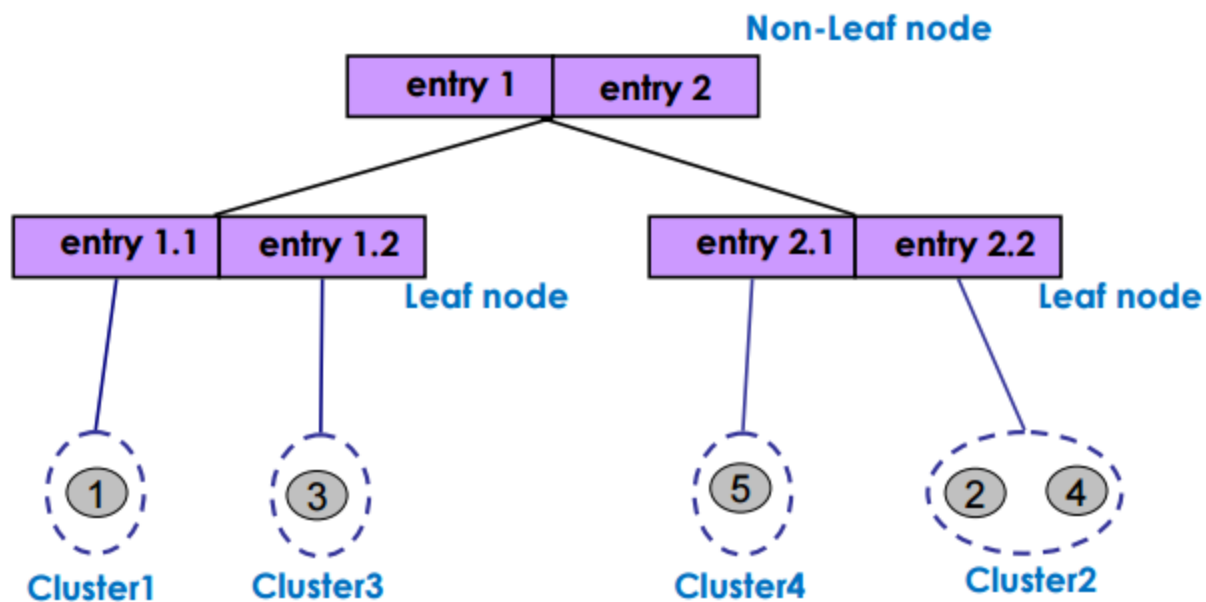
BUT there is a limit to the number of entries a node can have  
Thus, split the node

## BIRCH: The Idea by example

Data Objects

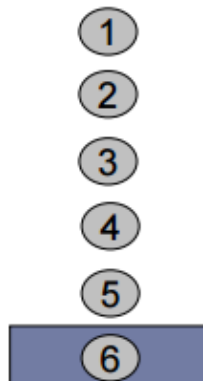


Clustering Process (build a tree)

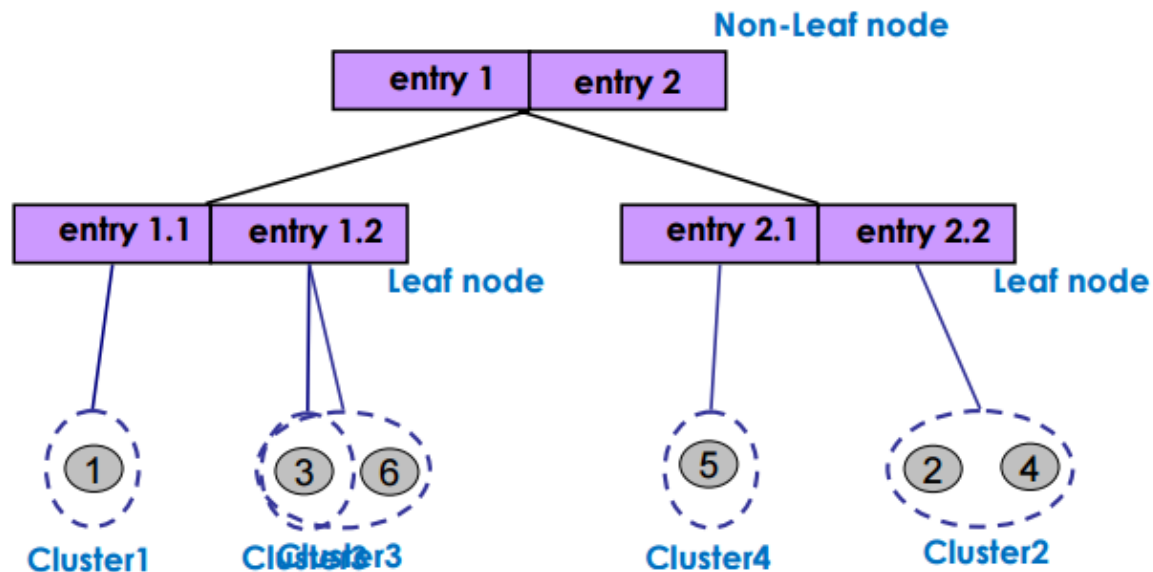


## BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)



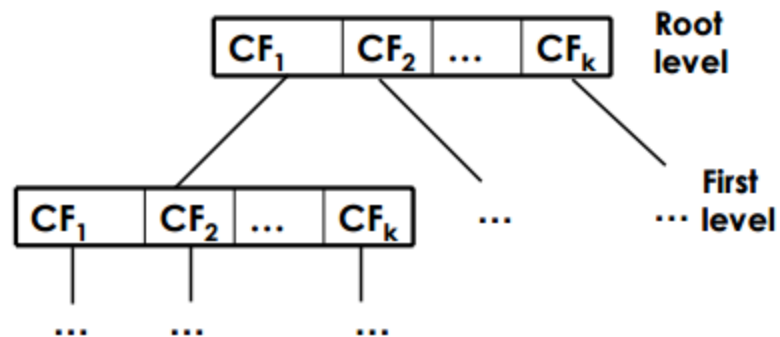
**entry1.2 is the closest to object 6**

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



## CF Tree

- ▶ **B** = Branching Factor, maximum children in a non-leaf node
- ▶ **T** = Threshold for diameter or radius of the cluster in a leaf
- ▶ **L** = number of entries in a leaf
- ▶ CF entry in parent = sum of CF entries of a child of that entry
- ▶ In-memory, height-balanced tree



## Clustering Feature

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

$$CF_3 = CF_1 + CF_2 = \langle 3+3, (9+35, 10+36), (29+417, 38+440) \rangle = \langle 6, (44, 46), (446, 478) \rangle$$

Cluster3

Cluster 1

(2,5)  
(3,2)  
(4,3)



Cluster 2



$$CF_2 = \langle 3, (35, 36), (417, 440) \rangle$$

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

# BIRCH concepts and terminology

---

## CF Tree

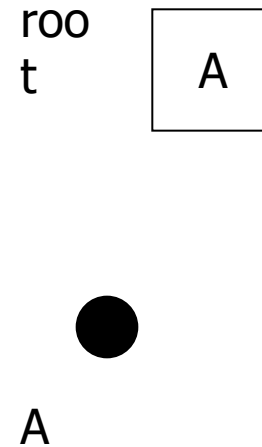
- The leaf contains actual clusters.
  - The size of any cluster in a leaf is not larger than  $T$ .
-

# BIRCH algorithm

---

- An example of the CF Tree

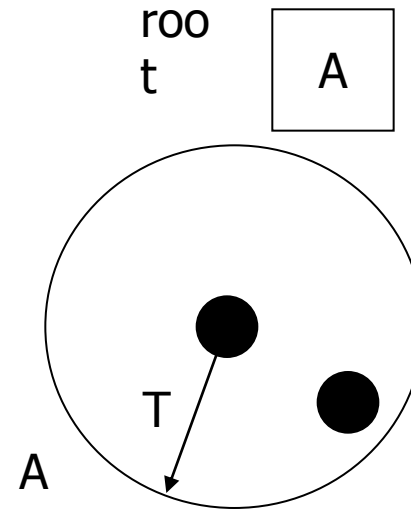
Initially, the data points in one cluster.



# BIRCH algorithm

- An example of the CF Tree

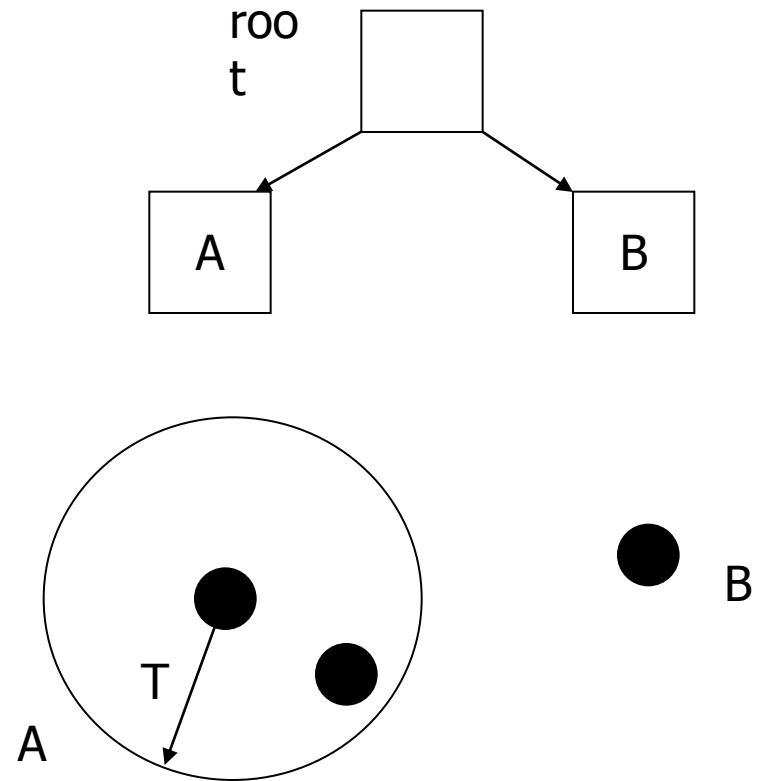
The data arrives, and a check is made whether the size of the cluster does not exceed  $T$ .



# BIRCH algorithm

- An example of the CF Tree

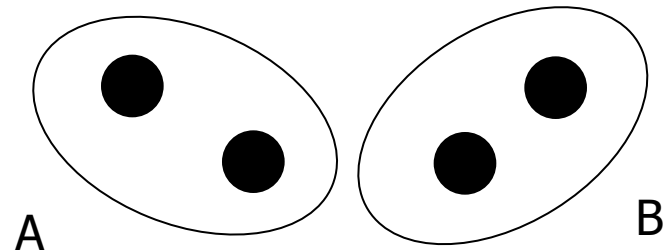
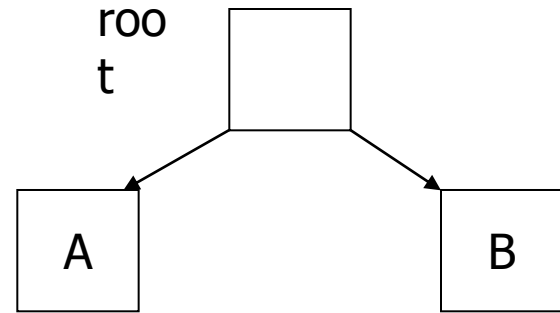
If the cluster size grows too big, the cluster is split into two clusters, and the points are redistributed.



# BIRCH algorithm

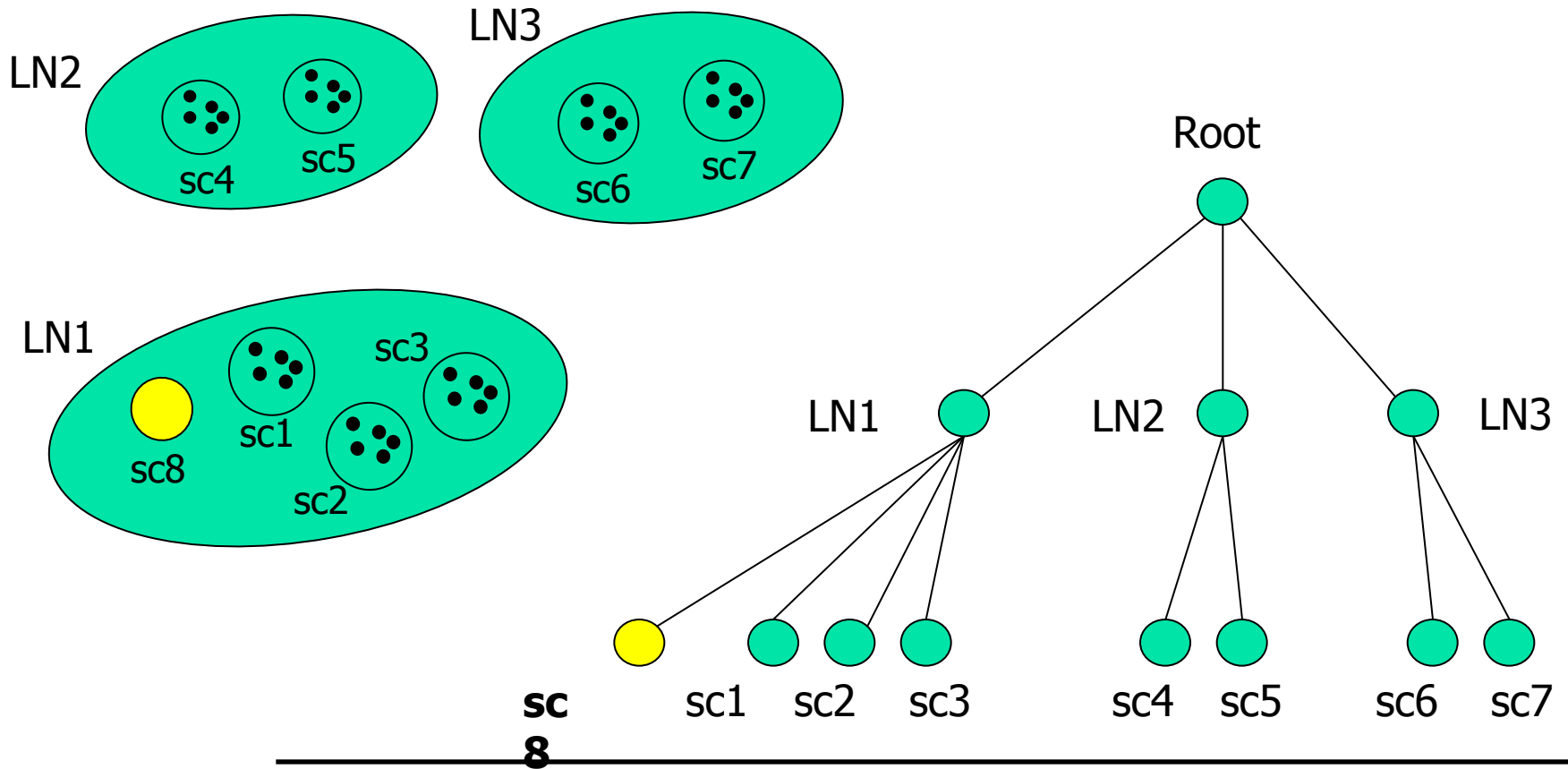
- An example of the CF Tree

At each node of the tree, the CF tree keeps information about the mean of the cluster, and the mean of the sum of squares to compute the size of the clusters efficiently.



# BIRCH algorithm

- Another example of the CF Tree Insertion

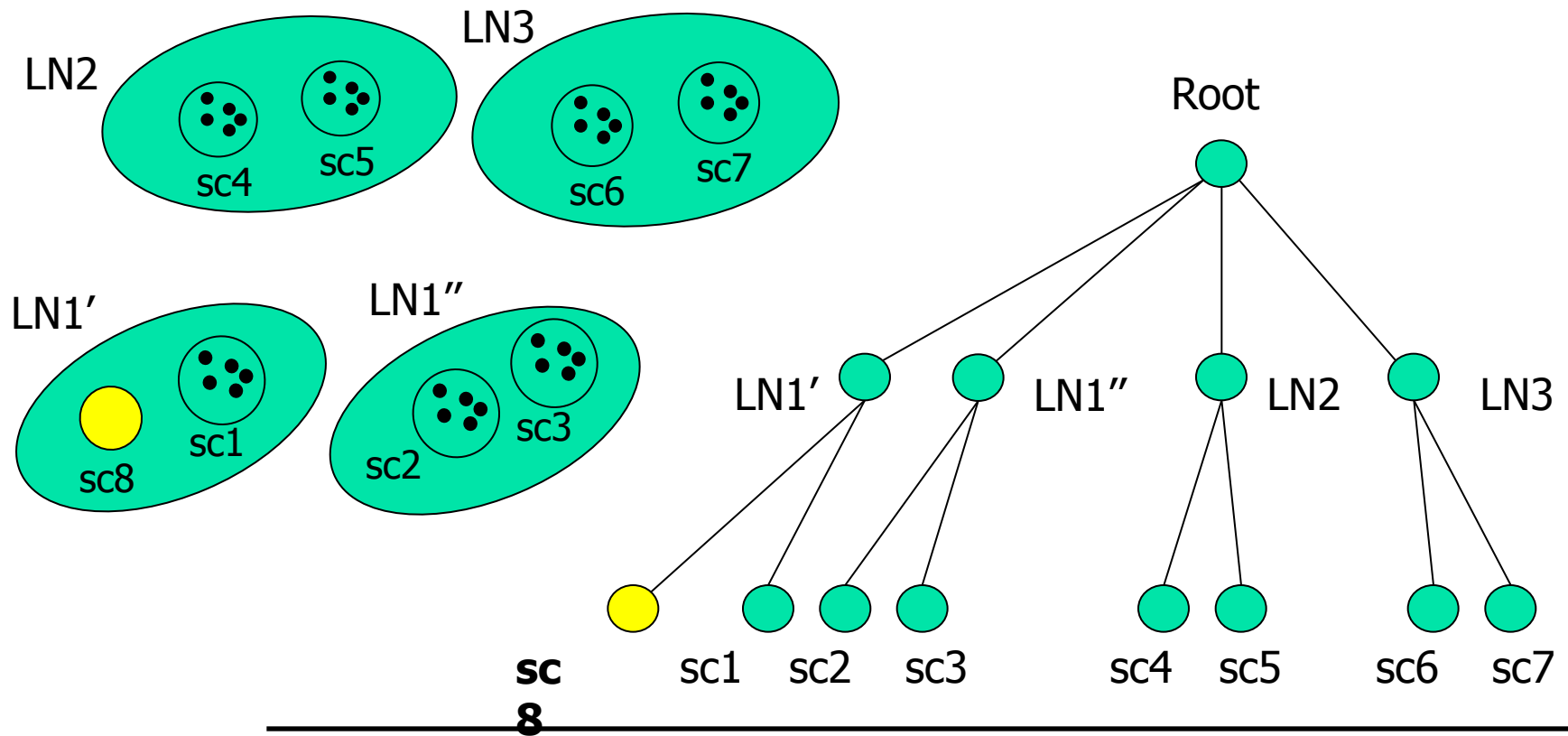




# BIRCH algorithm

- Another example of the CF Tree Insertion

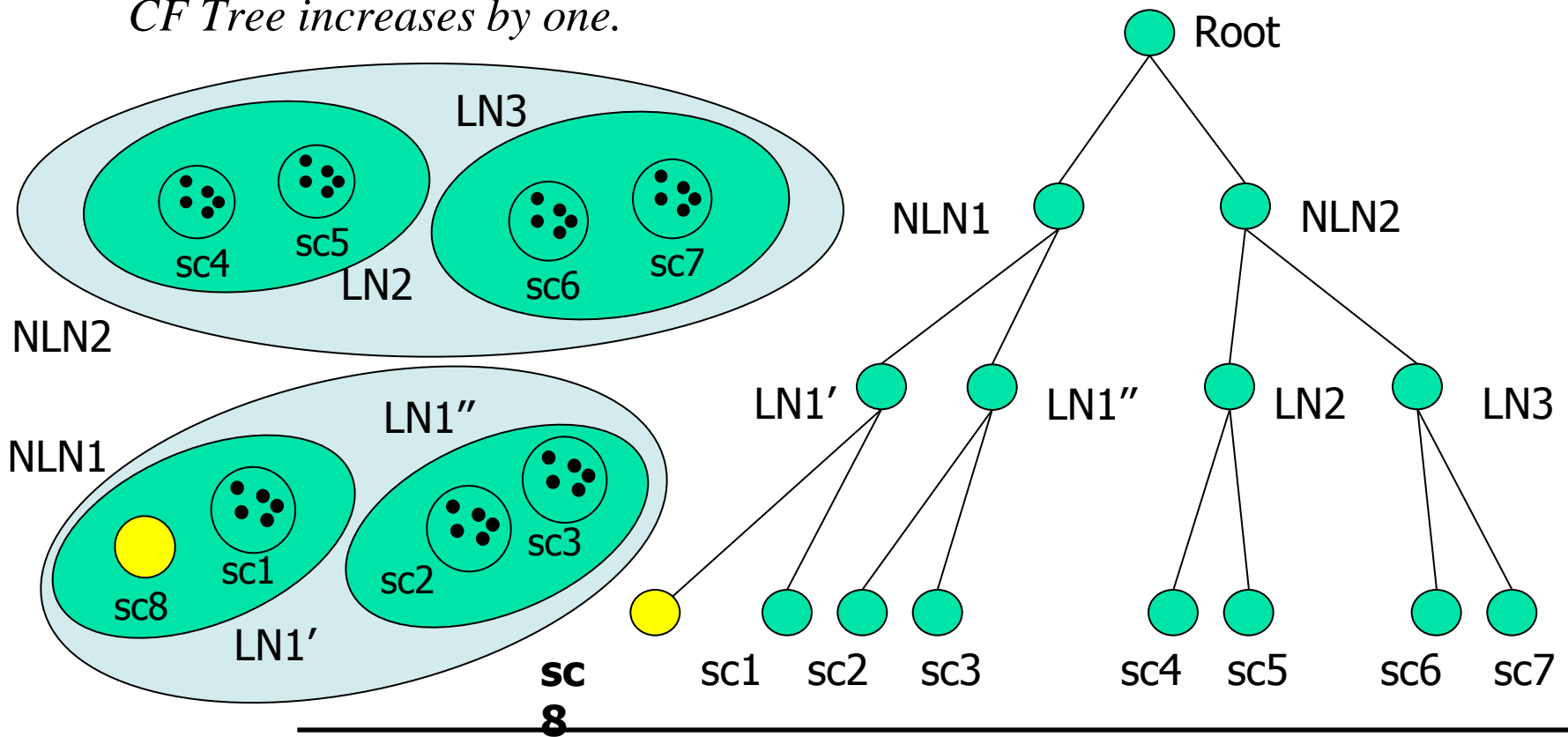
*If the branching factor of a leaf node can not exceed 3, then LN1 is split.*



## 5. BIRCH algorithm


- Another example of the CF Tree Insertion

*If the branching factor of a non-leaf node can not exceed 3, then the root is split and the height of the CF Tree increases by one.*



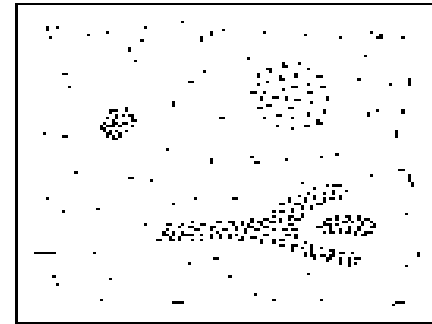
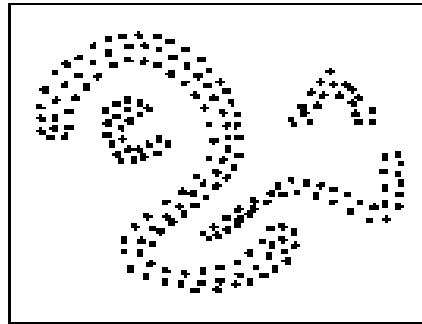
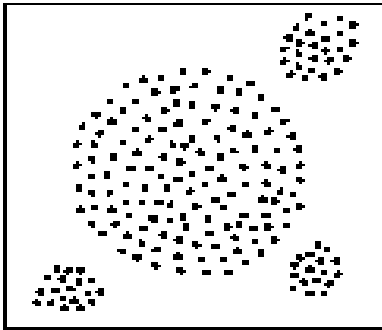
# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density based Methods 
- Introduction to grid based method
- Summary

# Density-based Approaches

---

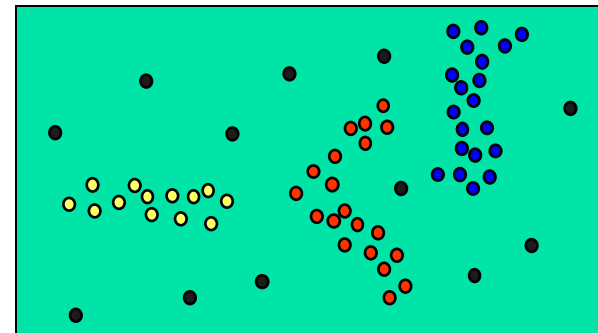


- Why Density-Based Clustering methods?
  - Discover clusters of arbitrary shape.
  - Clusters – Dense regions of objects separated by regions of low density

# Density-Based Clustering

- *Basic Idea:*

Clusters are dense regions in the data space, separated by regions of lower object density



Major features:

- Discover clusters of arbitrary shape

- Handle noise

- One scan

- Need density parameters

Several interesting studies:

- DBSCAN: Ester, et al. (KDD'96)

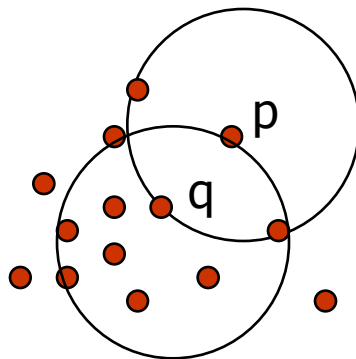
- DENCLUE: Hinneburg & D. Keim (KDD'98/2006)

- OPTICS: Ankerst, et al (SIGMOD'99).

- CLIQUE: Agrawal, et al. (SIGMOD'98)

# Density Concepts

- Two global parameters:
  - **Eps**: Maximum radius of the neighbourhood
  - **MinPts**: Minimum number of points in an Eps-neighbourhood of that point
- Core Object: object with at least MinPts objects within a radius ‘Eps-neighborhood’
- Border Object: object that on the border of a cluster



MinPts = 5

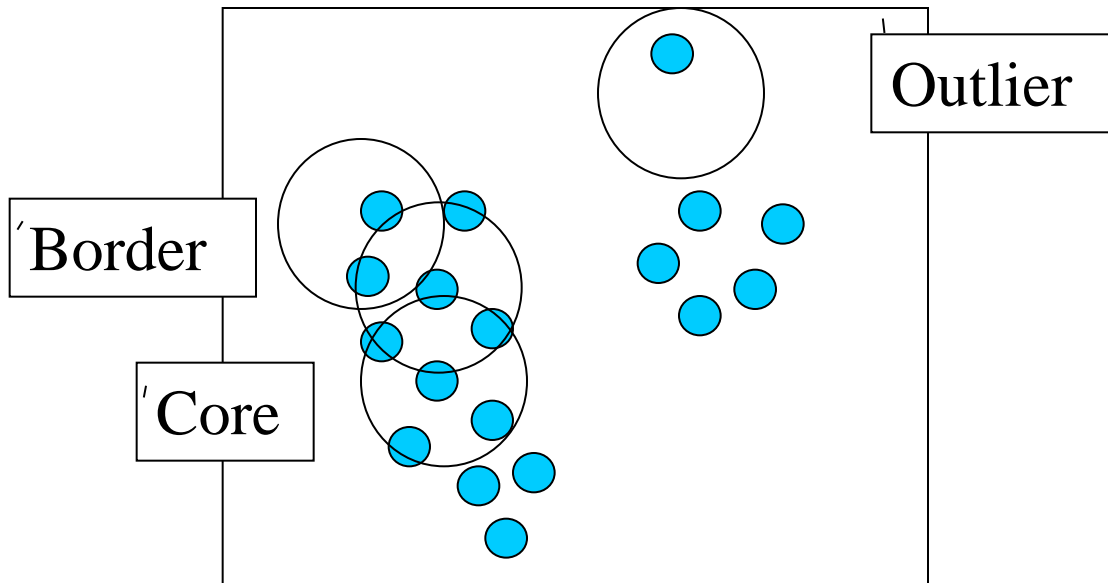
Eps = 1 cm

# DBSCAN

(<http://www2.cs.uh.edu/~ceick/7363/Papers/dbscan.pdf>)

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius  $r$  (Eps)
  - A point is a **core point** if it has more than a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
  - A **border point** has fewer than MinPts within Eps, but is in the neighborhood of a core point
  - A **noise point** is any point that is not a core point or a border point.

# Core, Border & Outlier



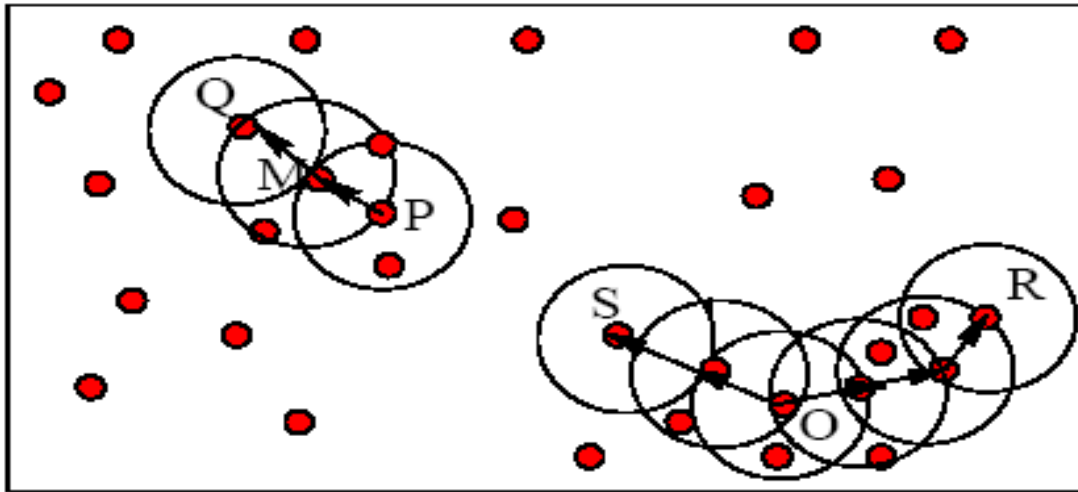
Given  $\epsilon$  and *MinPts*, categorize the objects into three exclusive groups.

$\epsilon = 1\text{unit}$ ,  $\text{MinPts} = 5$



# Example

- M, P, O, and R are core objects since each is in an Eps neighborhood containing at least 3 points



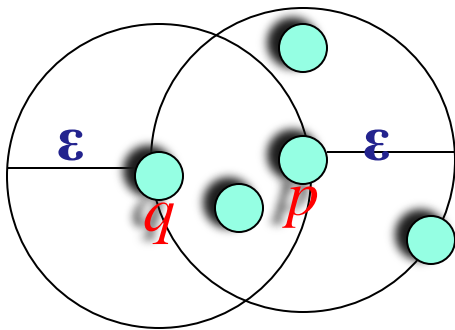
Minpts = 3

Eps=radius  
of the  
circles

# Density-Reachability

## □ Directly density-reachable

- An object  $q$  is directly density-reachable from object  $p$  if  $p$  is a core object and  $q$  is in  $p$ 's  $\epsilon$ -neighborhood.

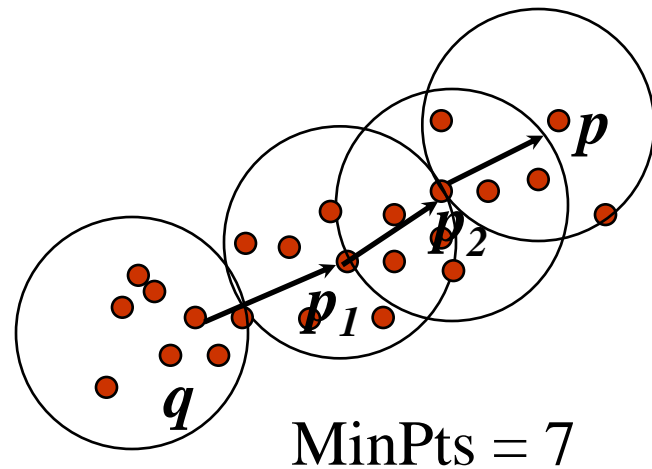


MinPts = 4

- $q$  is directly density-reachable from  $p$
- $p$  is not directly density-reachable from  $q$ ?
- Density-reachability is asymmetric.

# Density-reachability

- Density-Reachable (directly and indirectly):
  - A point  $p$  is directly density-reachable from  $p_2$ ;
  - $p_2$  is directly density-reachable from  $p_1$ ;
  - $p_1$  is directly density-reachable from  $q$ ;
  - $p \sqsubset p_2 \sqsubset p_1 \sqsubset q$  form a chain.



- $p$  is (indirectly) density-reachable from  $q$
- $q$  is not density-reachable from  $p$ ?

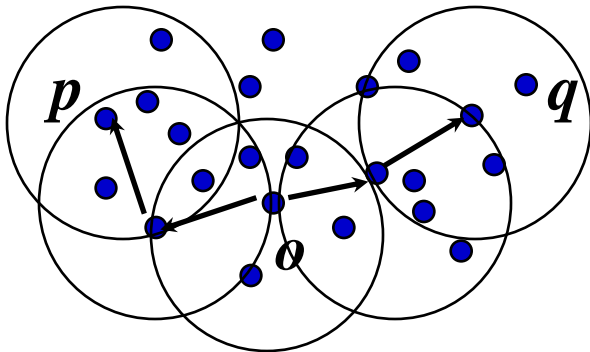
# Density-Connectivity

## □ Density-reachable is not symmetric

- not good enough to describe clusters

## □ Density-Connected

- A pair of points  $p$  and  $q$  are density-connected if they are commonly density-reachable from a point  $o$ .



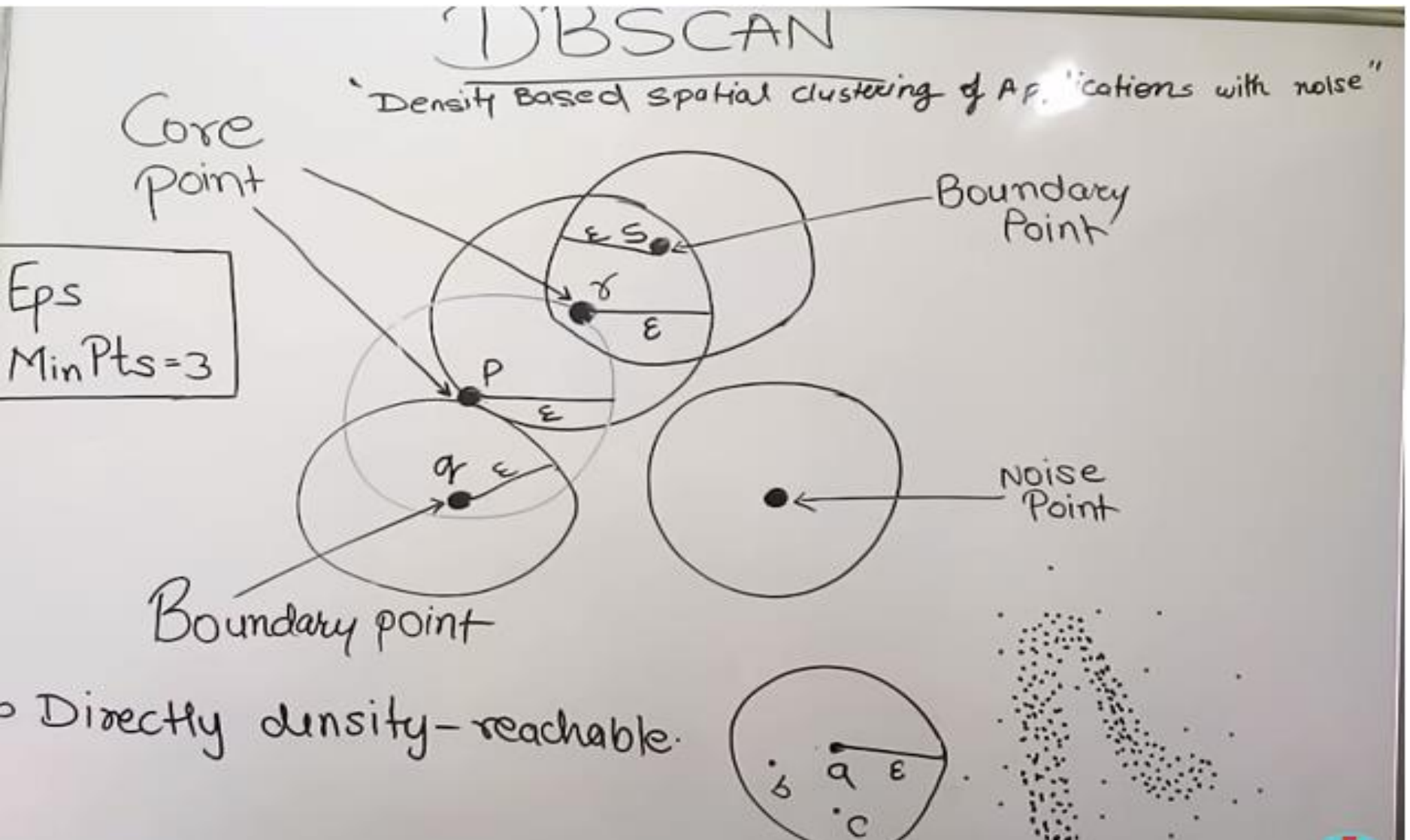
## □ Density-connectivity is symmetric

## DBSCAN: The Algorithm

---

- Arbitrary select a point  $p$
- Retrieve all points density-reachable from  $p$  wrt  $Eps$  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.

# Density-Based Clustering: DBSCAN

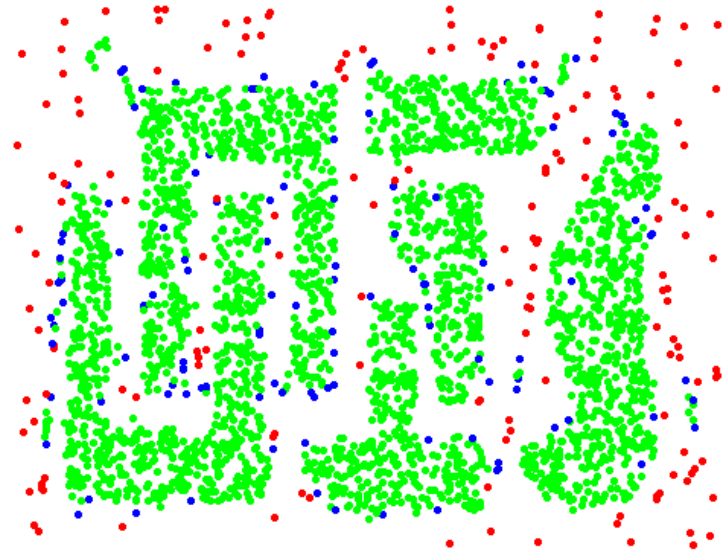


# Example

---



**Original Points**



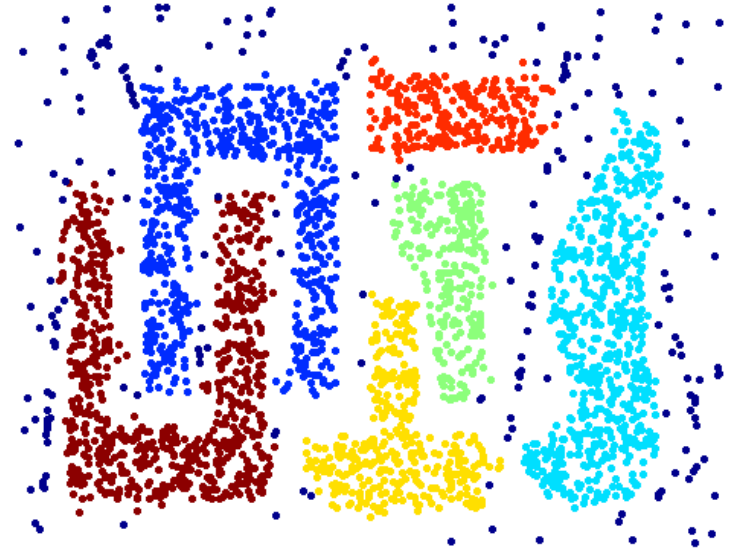
**Point types: core,  
border and outliers**

$\epsilon = 10$ , MinPts = 4

# When DBSCAN Works Well



**Original Points**

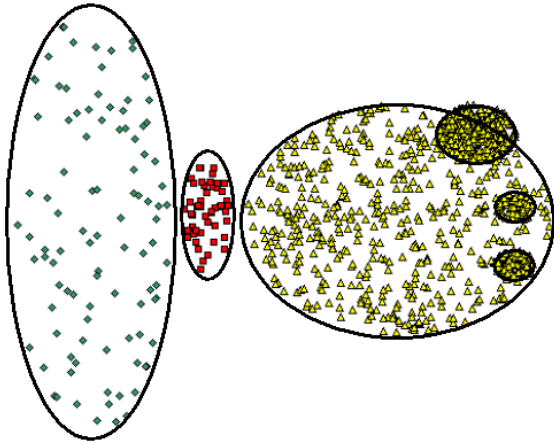


**Clusters**

- **Resistant to Noise**
- **Can handle clusters of different shapes and sizes**

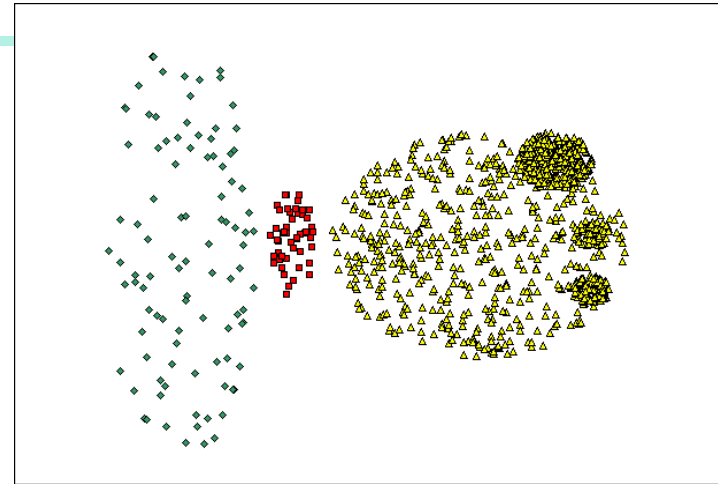


# When DBSCAN Does NOT Work Well

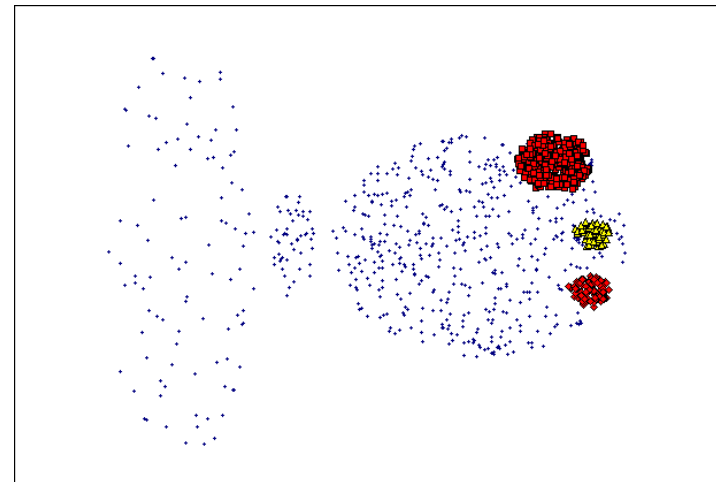


**Original Points**

- **Cannot handle Varying densities**
- **sensitive to parameters**




(MinPts=4, Eps=9.92).



(MinPts=4, Eps=9.75)

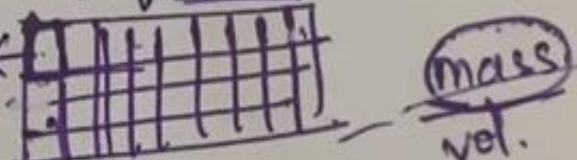
# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density based Methods
- Introduction to grid based method 
- Summary

## \* GRID BASED CLUSTERING: - grids

uses a multi resolution grid data structure.

it divides the object into finite no. of cells that form a grid like structure. cell.  mass  
vol.

- then density is calculated for these cells.
- Sort the cells according to density
- identify cluster centers
- update neighbour cells

\* Quick processing time

Ex: STING

(Statistical Information Grid clustering Algorithm)

- spatial data is divided into rectangular cells at different levels of resolution, these cells form a tree structure

Cells at higher level - contains smaller cells compared to its lower levels.


clustering - done based on parameters.

(mean, count, min, max, SD, type of dist) <sup>→ normal, r.</sup>

- calculation of these parameters should start at root and go down till bottom layer

# Chapter 10. Cluster Analysis: Basic Concepts and Methods

---

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density based Methods
- Summary 

# Determine the Number of Clusters

---

- Empirical method
  - # of clusters:  $k \approx \sqrt{n}/2$  for a dataset of  $n$  points, e.g.,  $n = 200$ ,  $k = 10$
- Other methods:
  - Elbow method
  - Cross validation method

# Summary

---

- **Cluster analysis** groups objects based on their **similarity** and has wide applications
- 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