More clustering algorithms

Lecture 05.02

Clustering algorithms

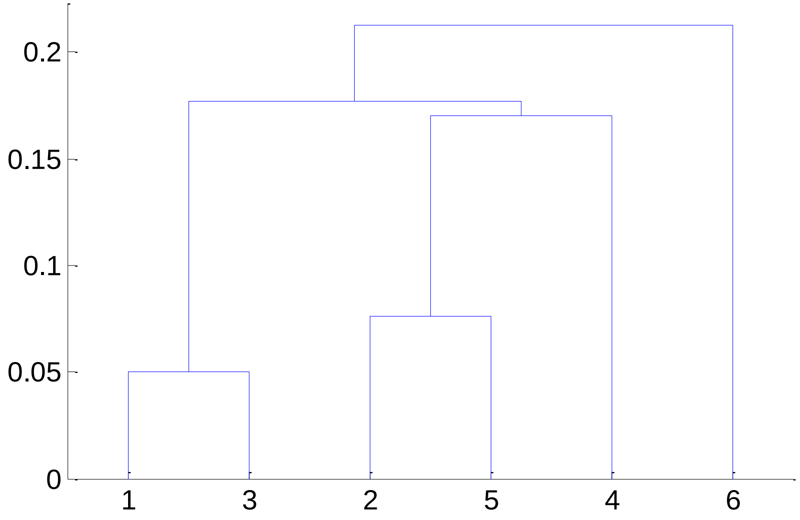
- *K*-means clustering
- Agglomerative hierarchical clustering
- Density-based clustering

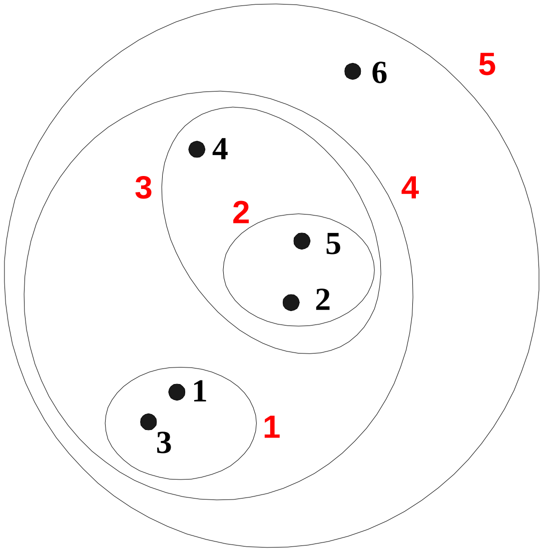
Clustering algorithms

- *K*-means clustering
- Agglomerative hierarchical clustering
- Density-based clustering

Hierarchical Clustering

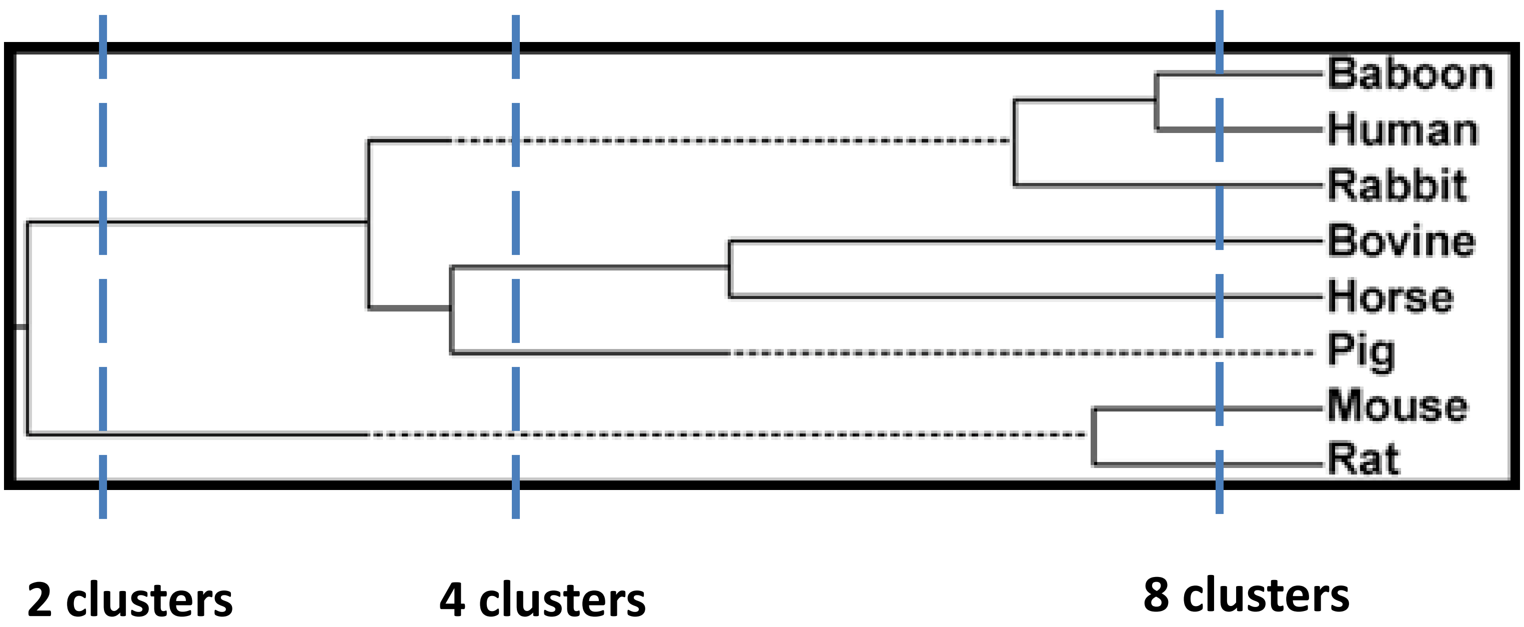
- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a *dendrogram*
  - A tree-like diagram that records the sequences of merges or splits





Strengths of hierarchical clustering

- Do not have to assume any particular number of clusters
  - 'cut' the dendogram at the proper level

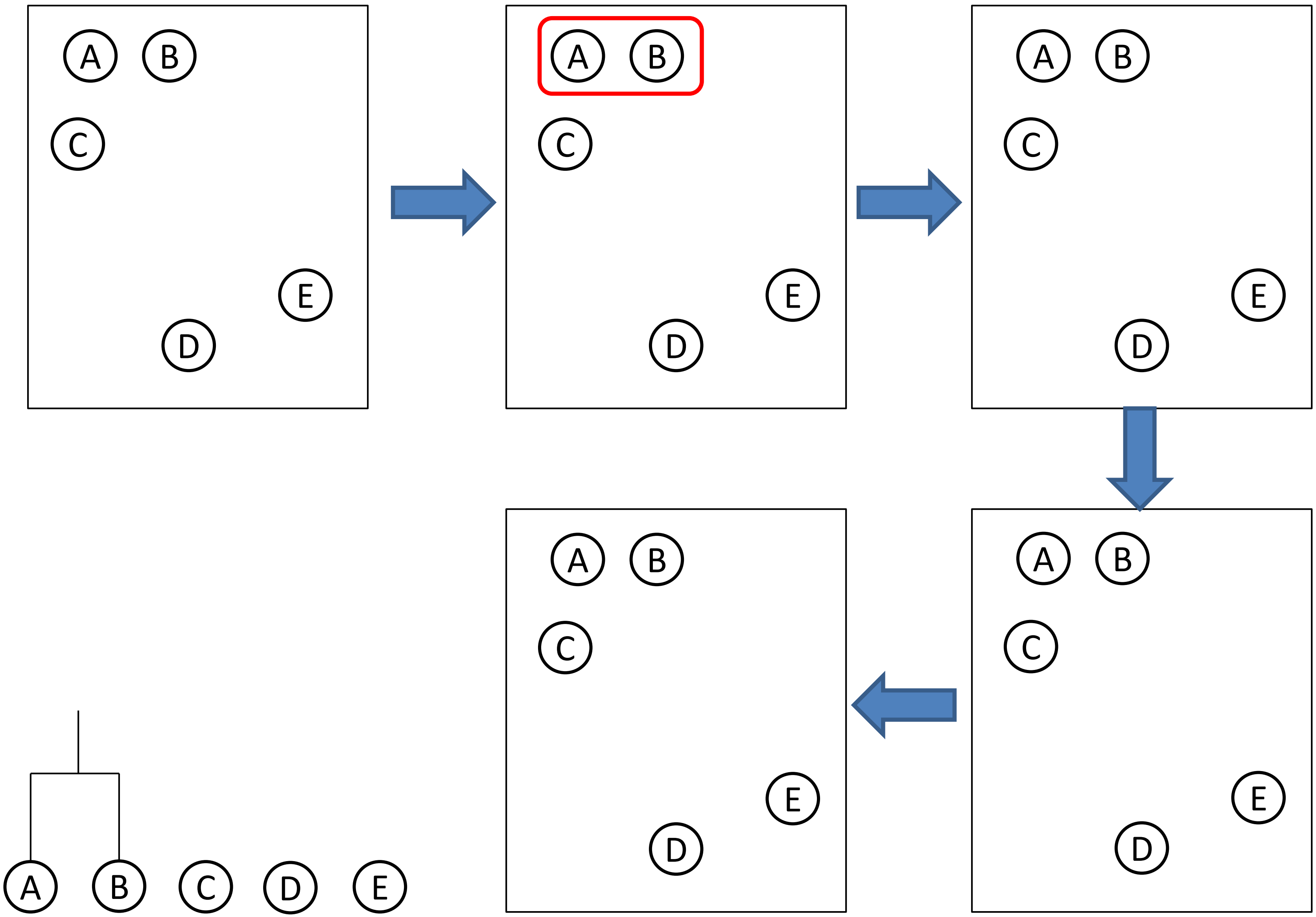


Types of hierarchical clustering

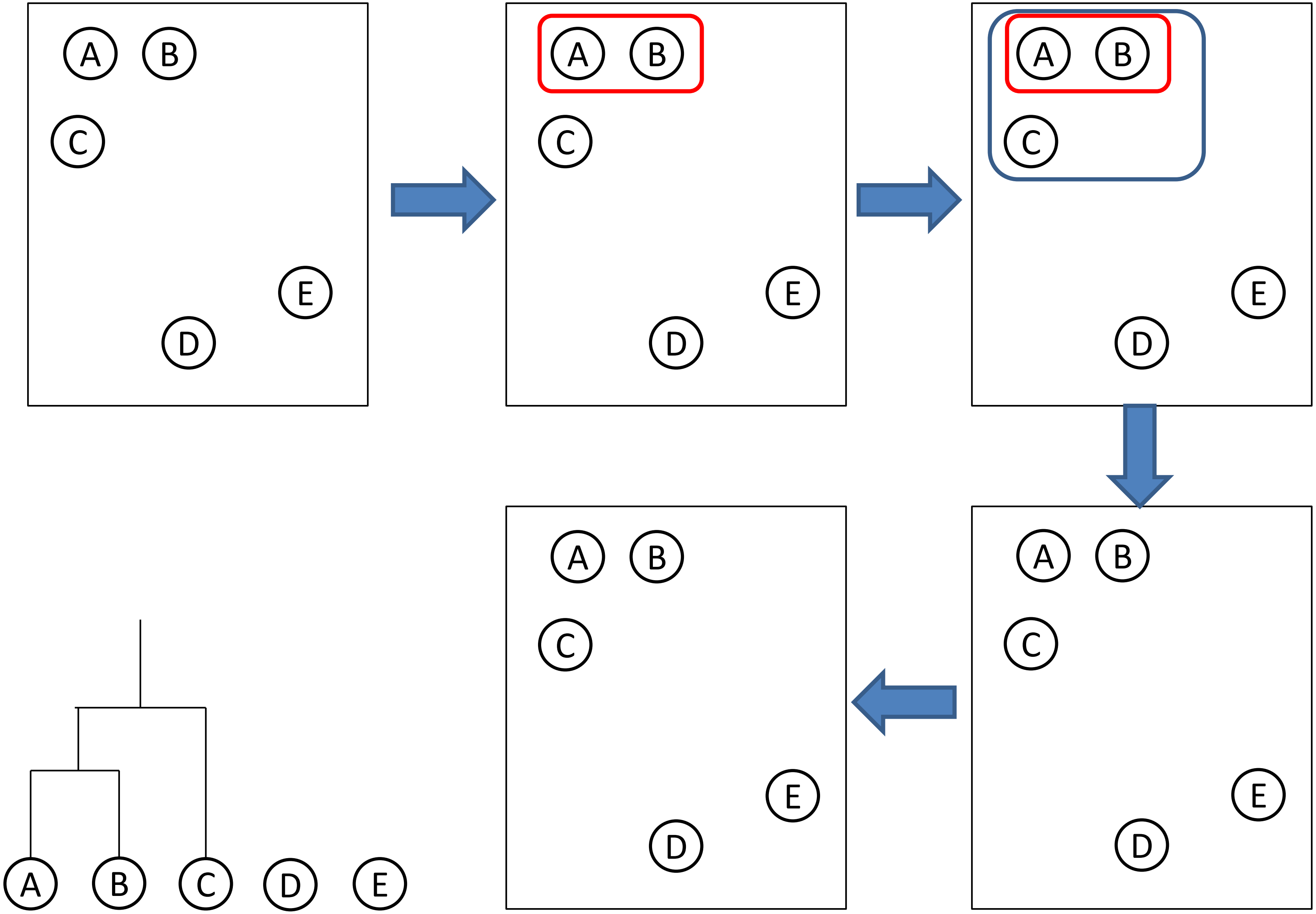
- *Agglomerative* – starts with each point as a cluster, and performs successive merges

- *Divisive* – starts with all points as a cluster and performs successive splits

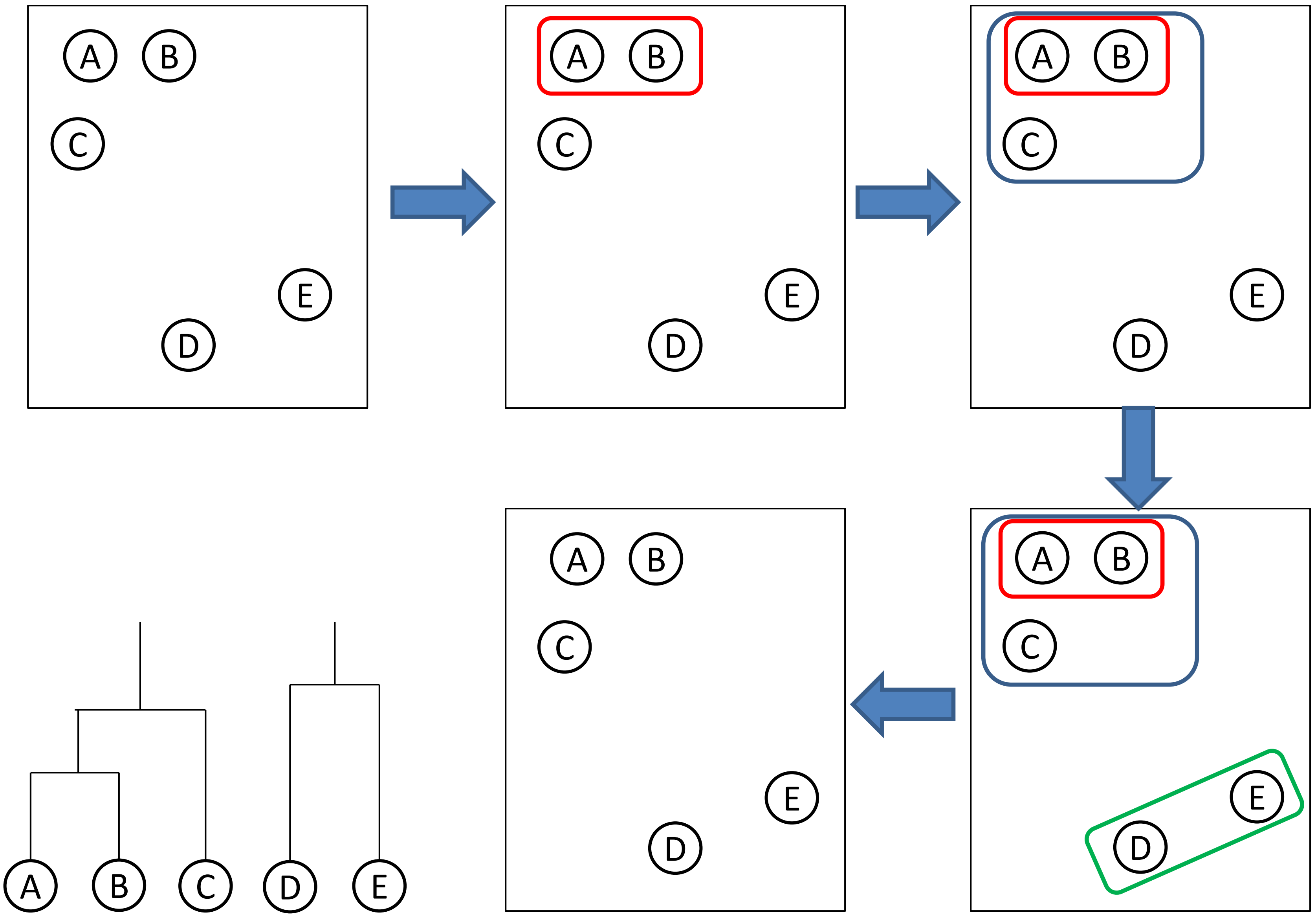
Hierarchical clustering example



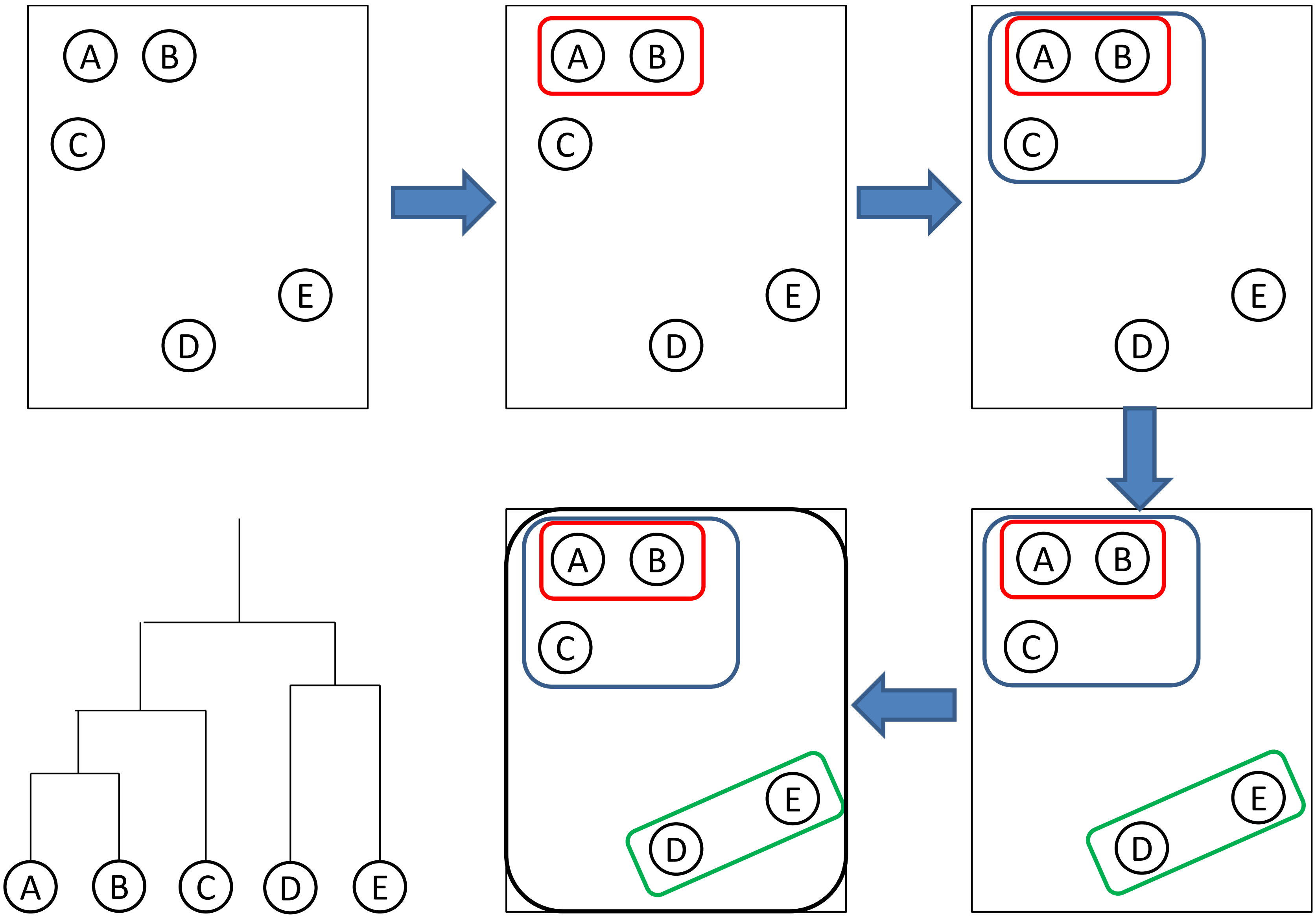
Hierarchical clustering example



Hierarchical clustering example



Hierarchical clustering example



Hierarchical Clustering Algorithm

- Start with the points as individual clusters
- At each step, merge the closest pair of clusters until only one cluster left.

Hierarchical Clustering Algorithm

Let each data point be a cluster
Compute the proximity matrix
Repeat
    Merge the two closest clusters
    Update the proximity matrix
Until only a single cluster remains

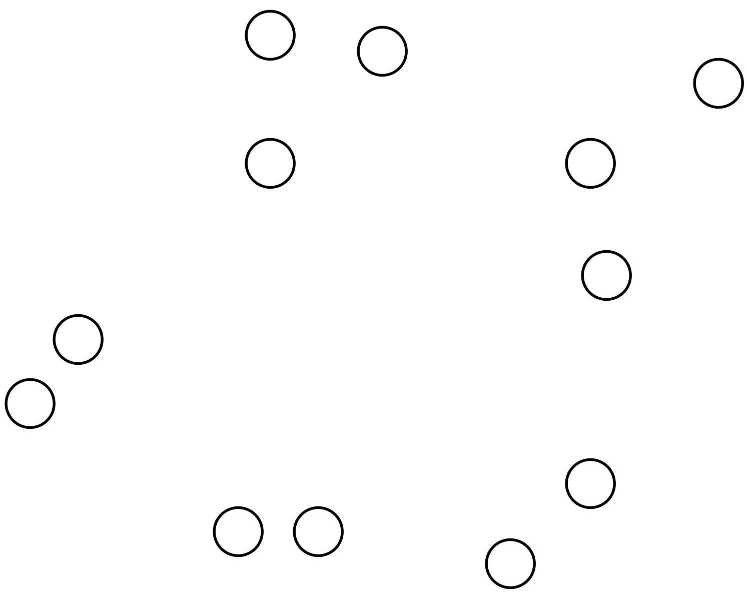
- Key operation is the computation of the proximity of two clusters.

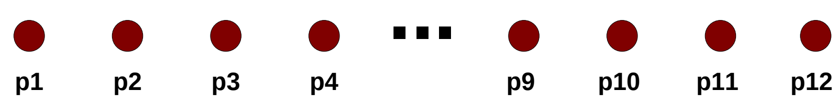
Starting Situation

- Start with clusters of individual points and a proximity matrix

	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

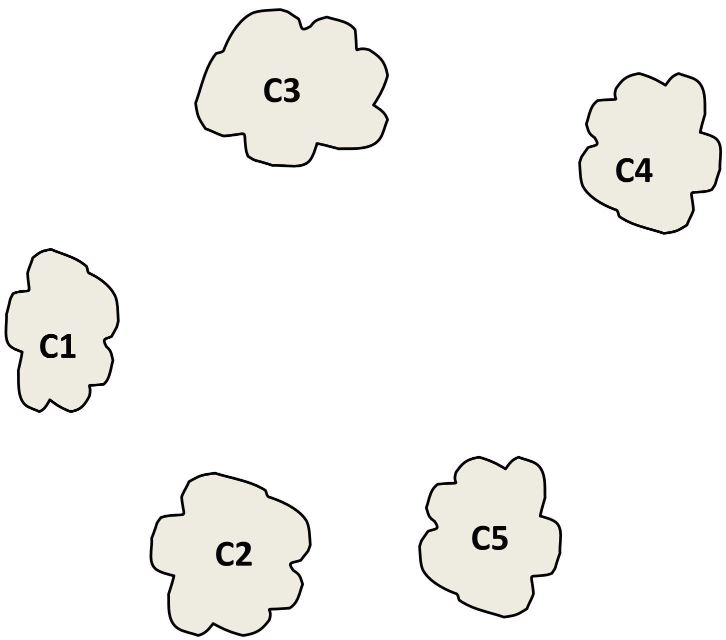
Proximity Matrix





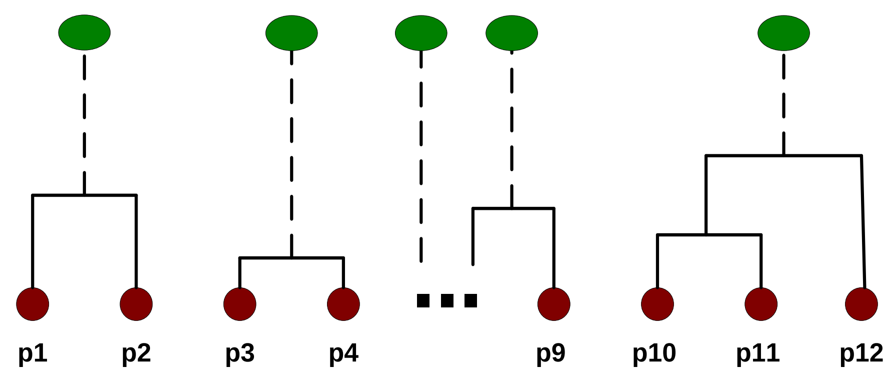
Intermediate Situation

- After some merging steps, we have some clusters



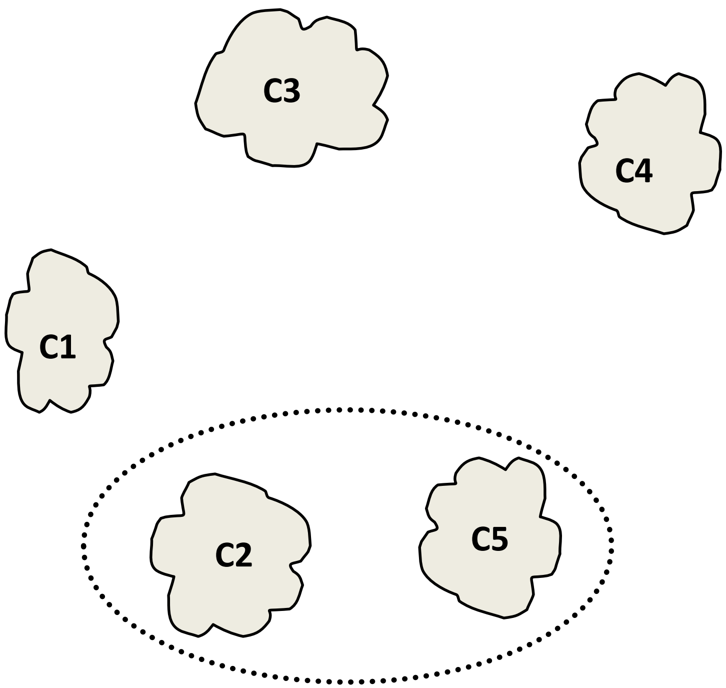
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

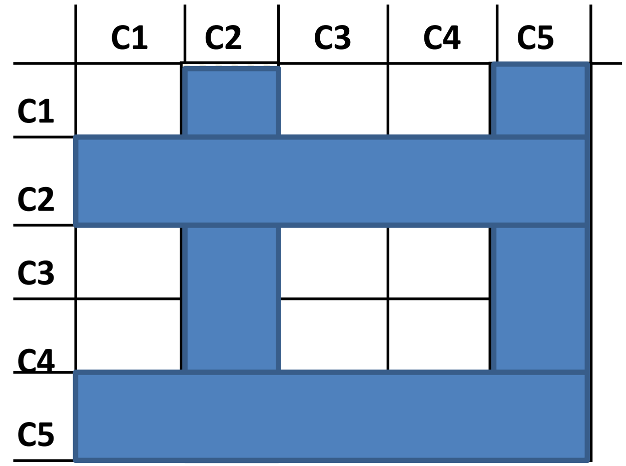
Proximity Matrix



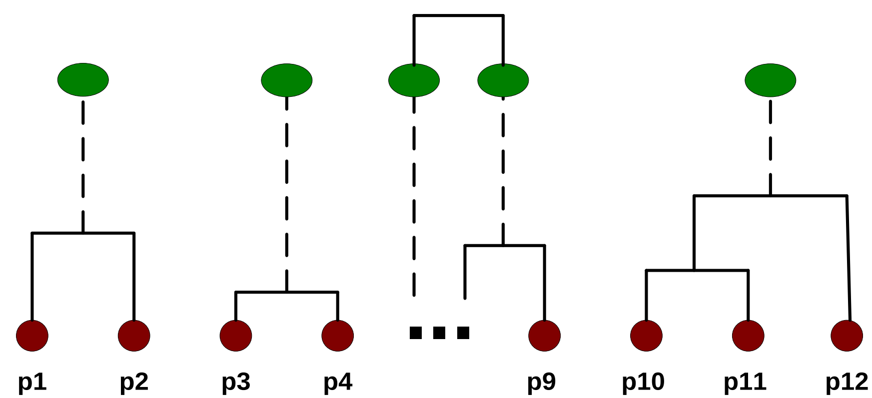
Intermediate Situation

- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.



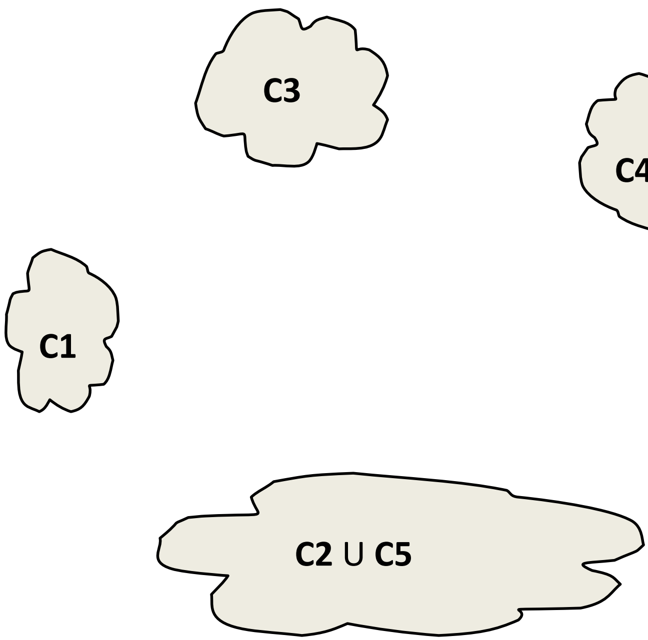


Proximity Matrix



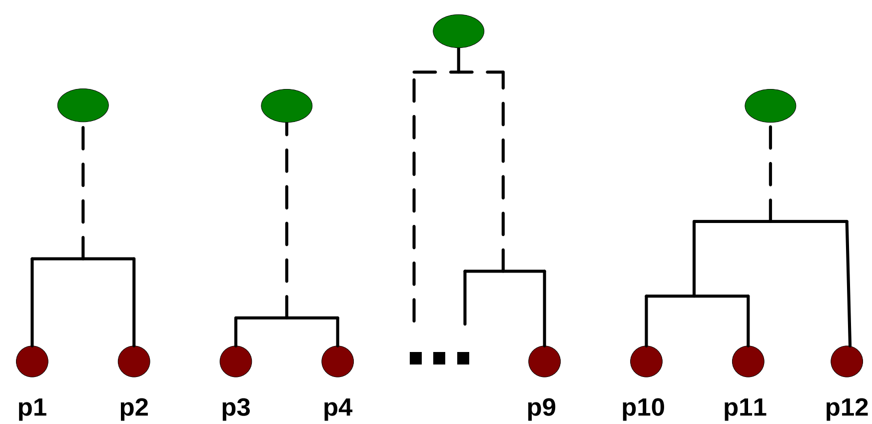
After Merging

- The question is "How do we update the proximity matrix?"

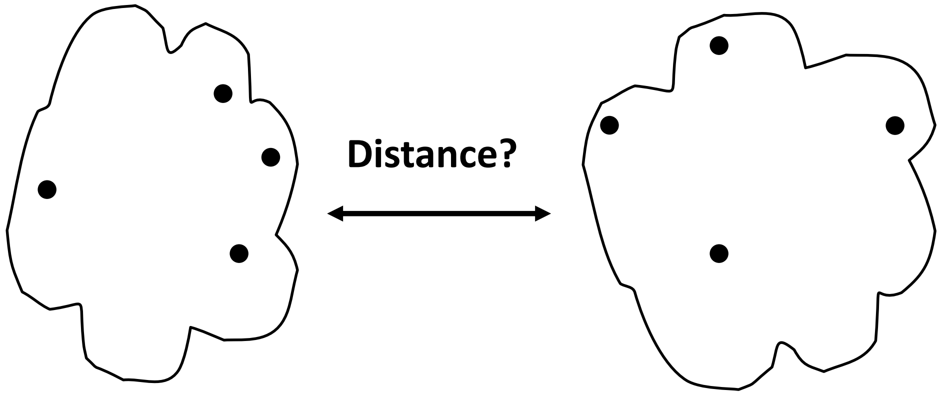


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?		?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Distance

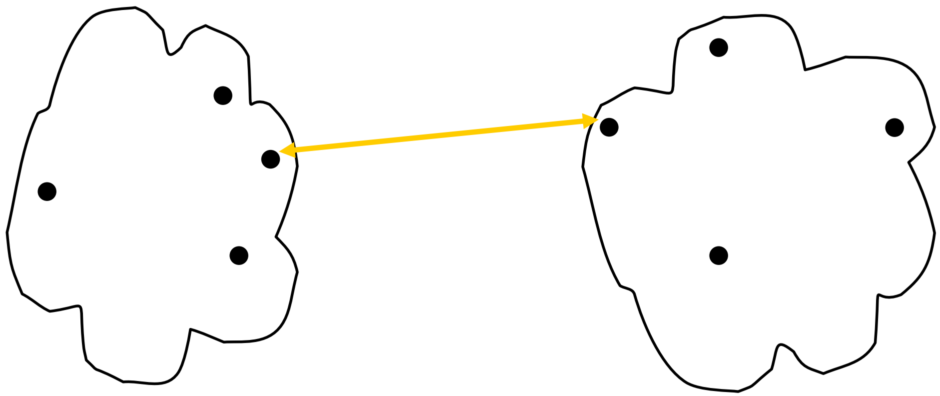


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Centroids Distance
- Group Average

Inter-Cluster Distance: MIN

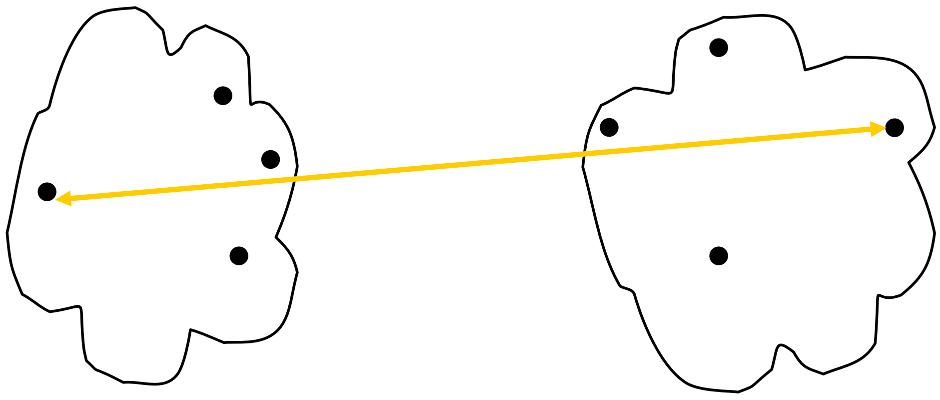


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

Problem: sensitive to outliers

Inter-Cluster Distance: MAX

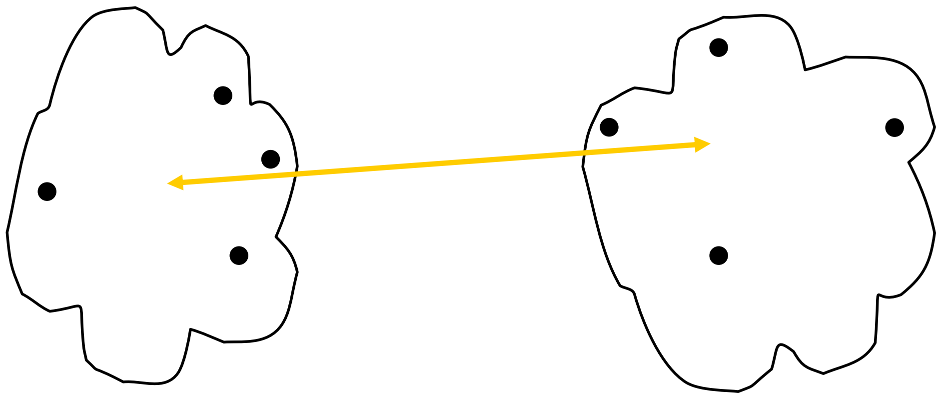


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

Problem: tends to break large clusters

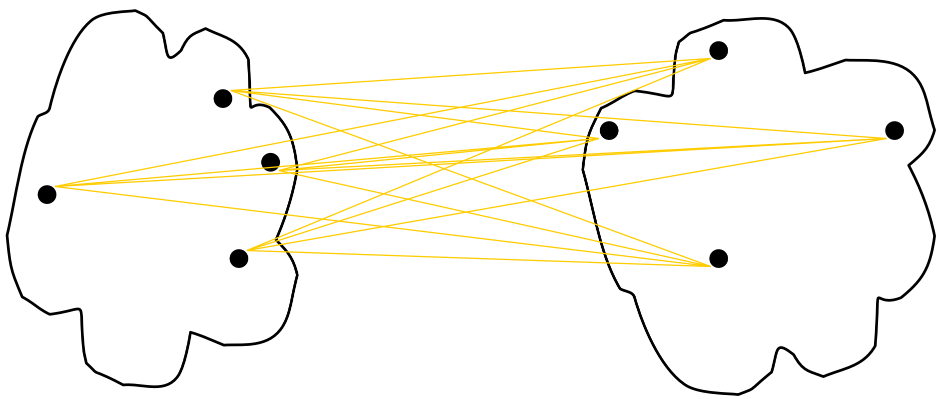
Inter-Cluster Distance: Centroid distance



	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

Inter-Cluster Distance: Group Average

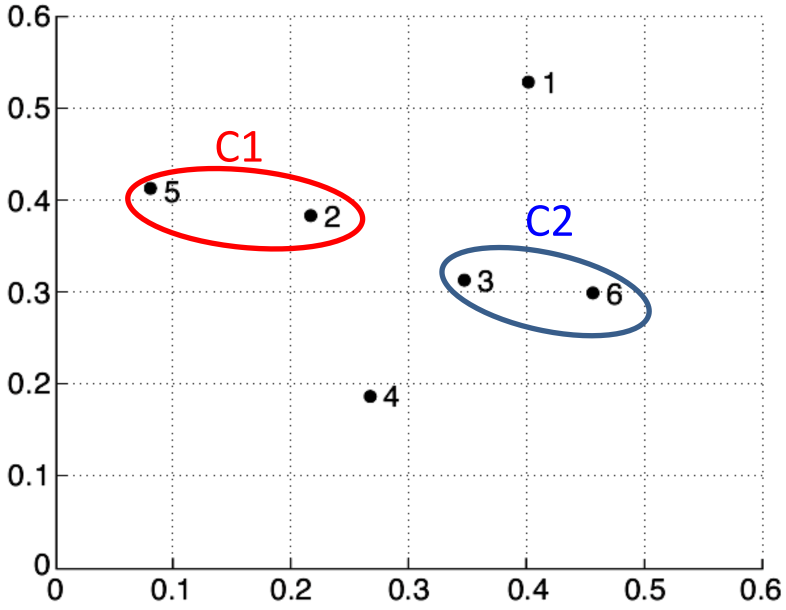


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

Cluster Distance: MIN (single link)

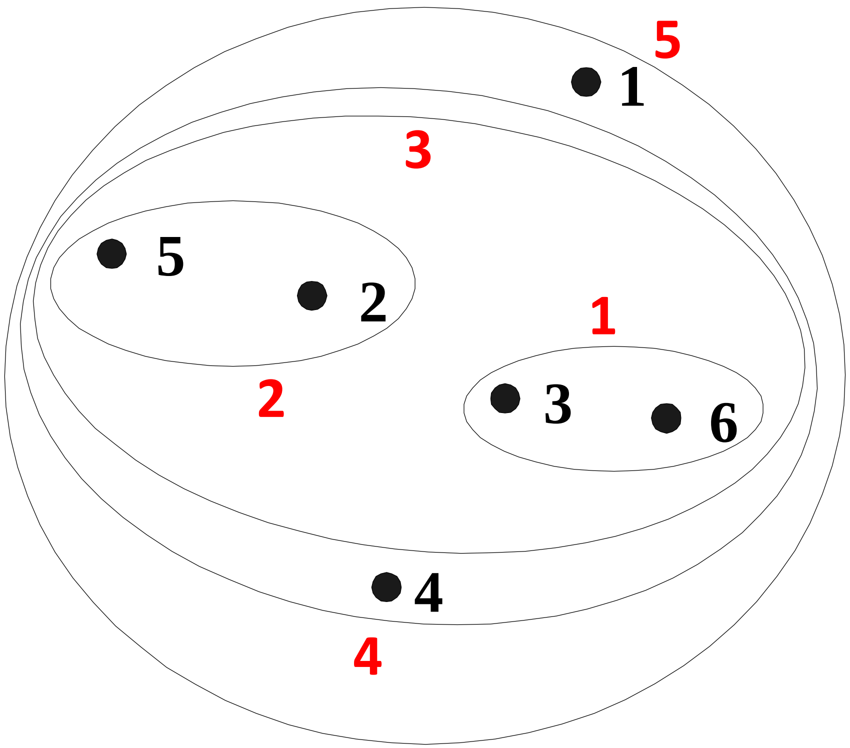
- Distance between two clusters is based on the two most similar (closest) points in the different clusters
  - Determined by one pair of points



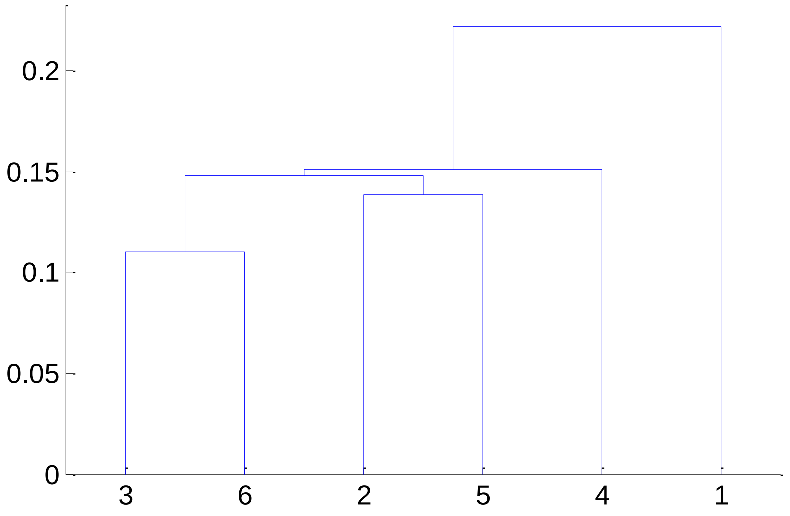
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

$d(\mathrm{C1},\mathrm{C2})=0.15$

Hierarchical Clustering: MIN



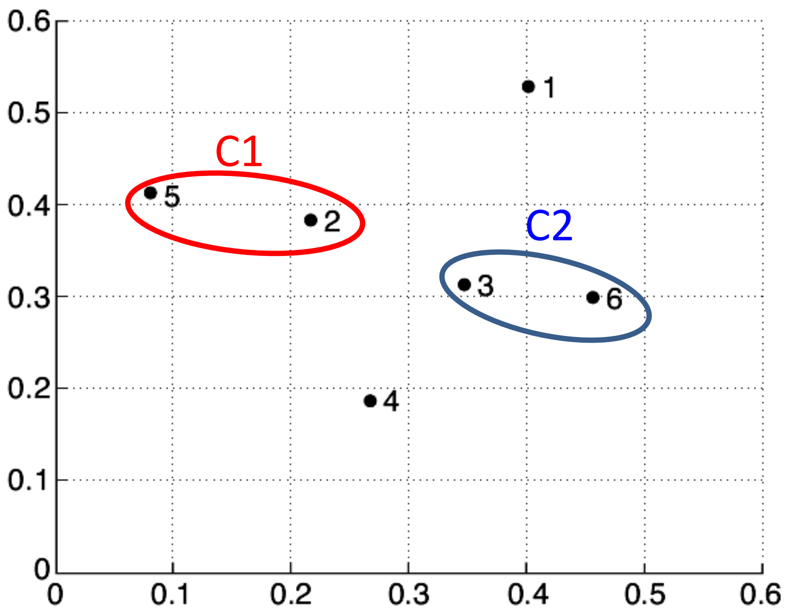
Nested Clusters



Dendrogram

Cluster Distance: MAX

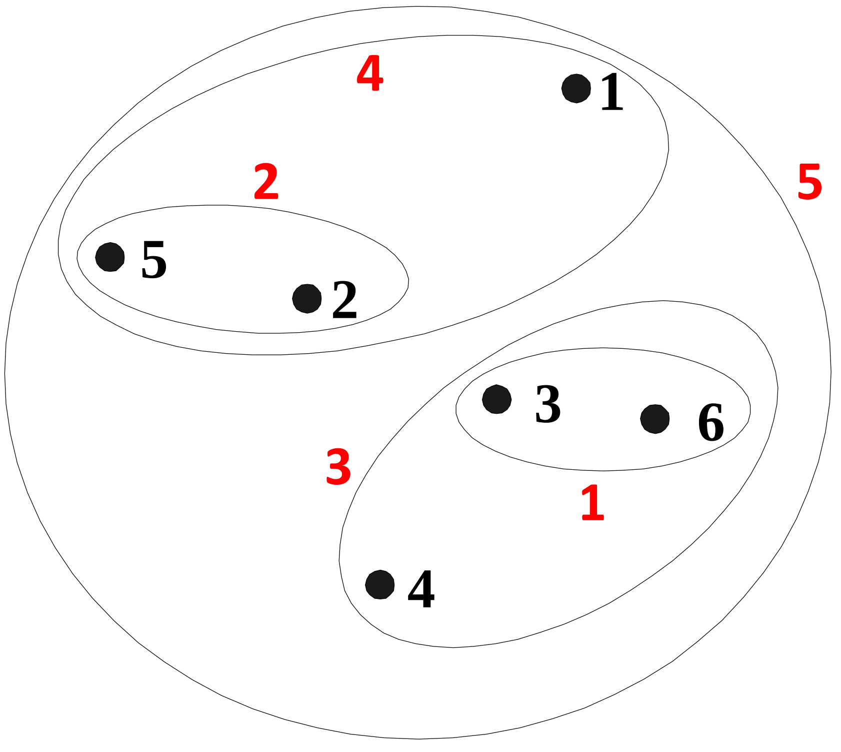
- Distance between two clusters is based on the two least similar (most distant) points in the different clusters
  - Determined by one pair of points



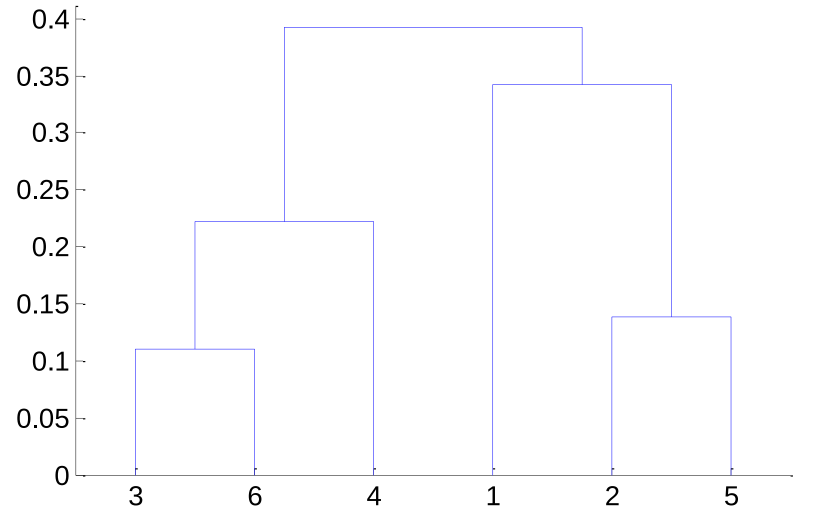
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

$d(\mathrm{C1},\mathrm{C2})=0.39$

Hierarchical Clustering: MAX



Nested Clusters



Dendrogram

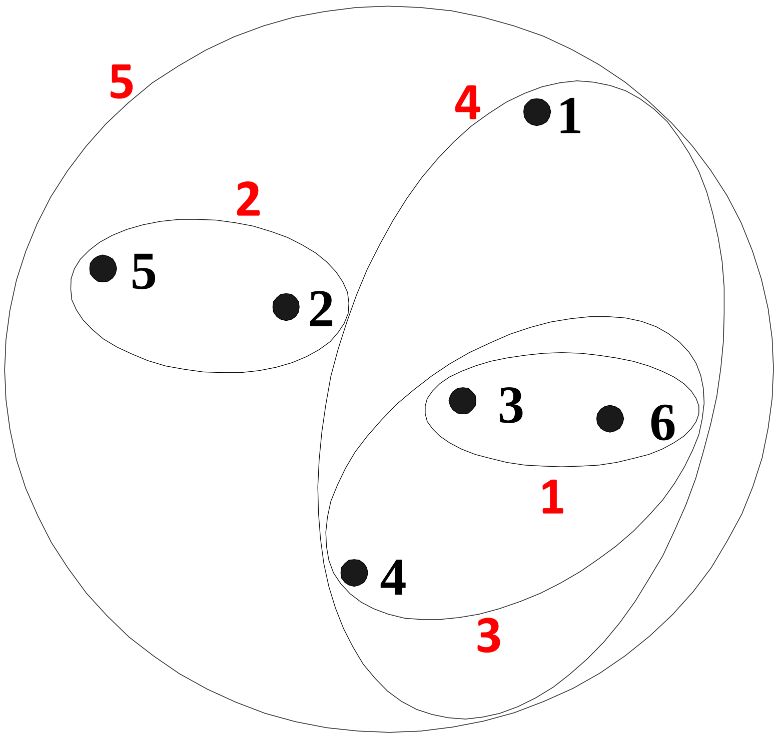
Hierarchical clustering: Group Average

- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

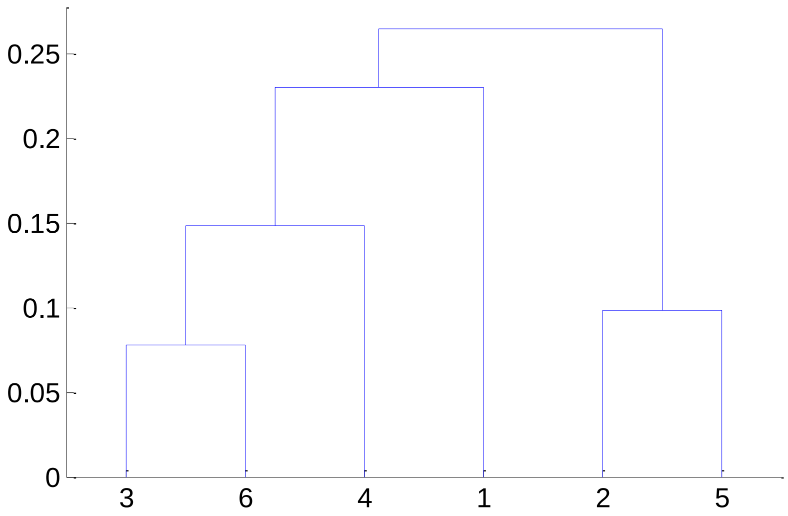
$$\mathbf{proximity}(\mathbf{Cluster_i}, \mathbf{Cluster_j}) = \frac{\sum\limits_{\substack{p_i \in Cluster_i \\ p_j \in Cluster_j}} \mathbf{proximity}(\mathbf{p_i}, \mathbf{p_j})}{|\mathbf{Cluster_i}| * |\mathbf{Cluster_j}|}$$

  – uses all pairs of points from two clusters

Cluster distance: Group Average



Nested Clusters

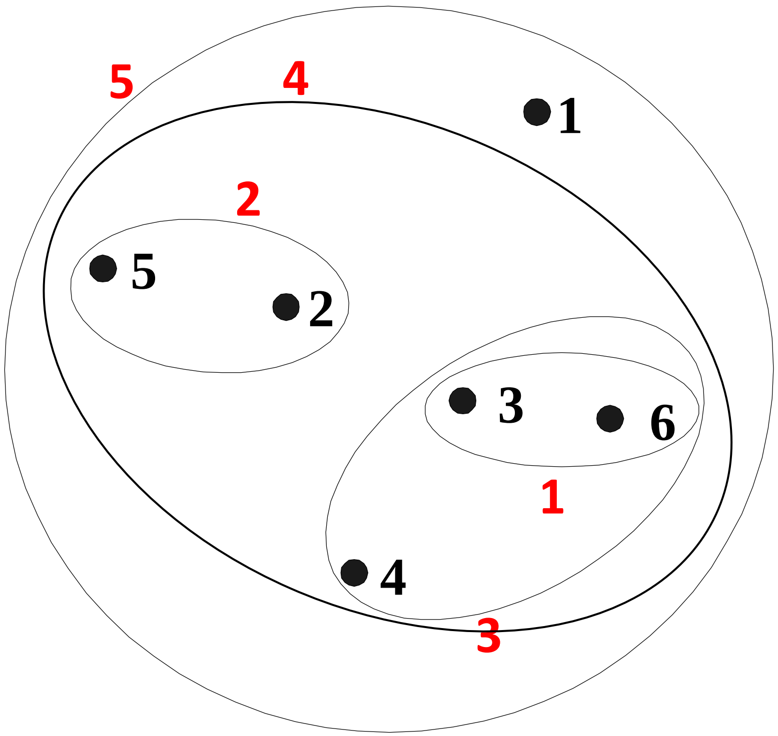


Dendrogram

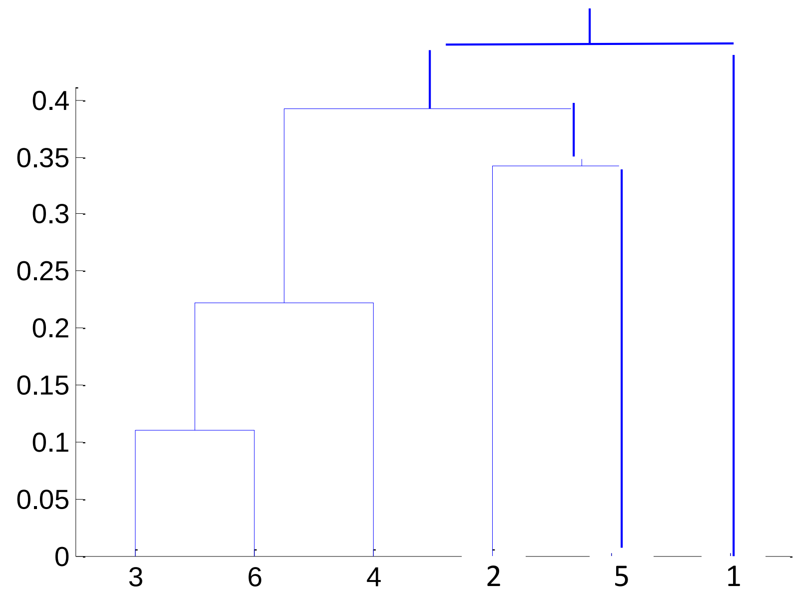
Cluster Distance: Centroid distance

- Distance between two clusters is based on the distance between their centroids
  - Determined by all points in each cluster

Cluster distance: Centroid distance



Nested Clusters



Dendrogram

Hierarchical Clustering: Time and Space

- $O(N^2)$ space since it uses the proximity matrix.
  - N is the number of points.

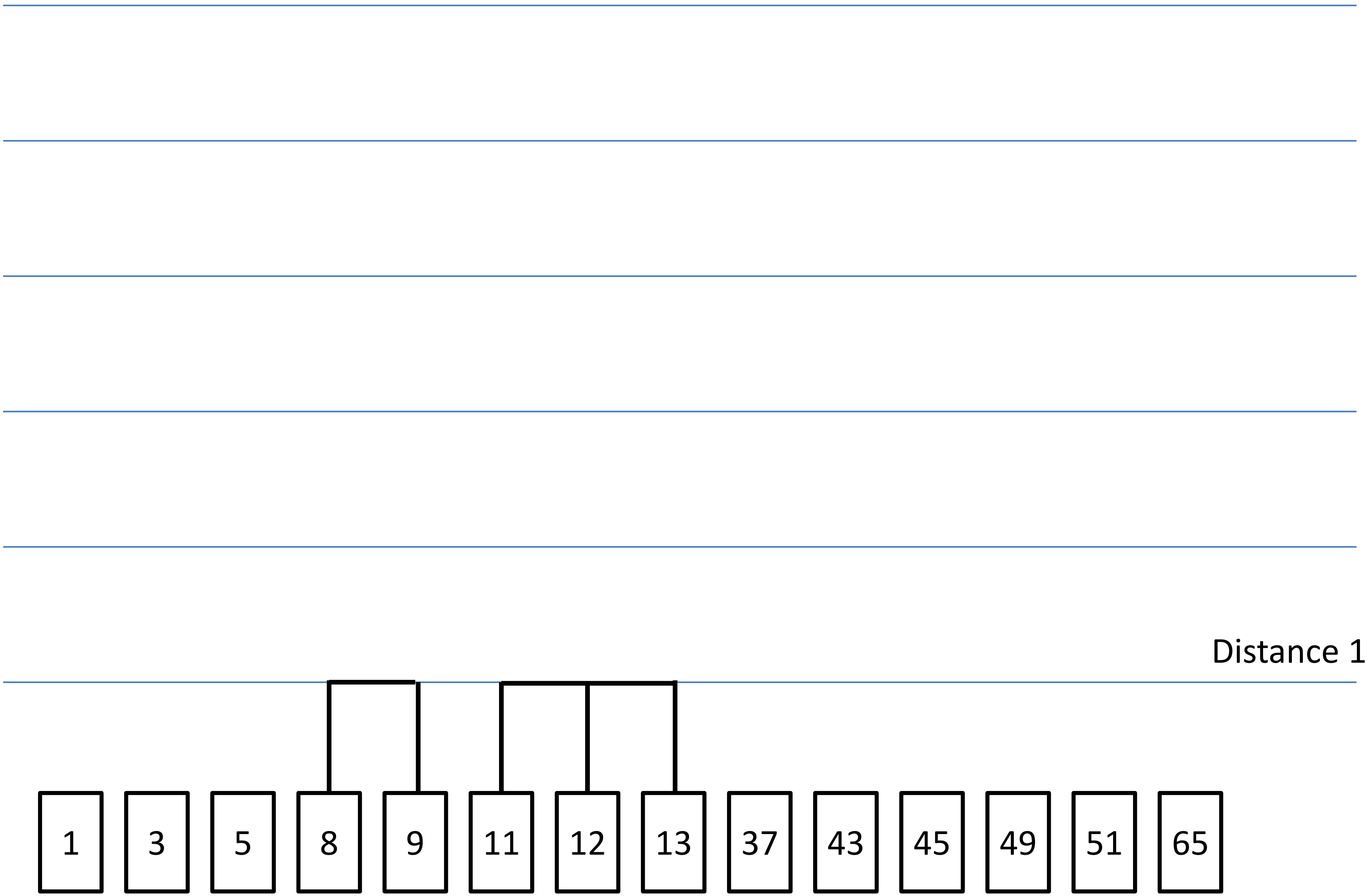
- $O(N^3)$ time in many cases
  - There are N steps and at each step the size, N^2, proximity matrix must be updated and searched
  - Complexity can be reduced to $O(N^2 \log(N))$ time using more advanced data structures

Hierarchical clustering is expensive !

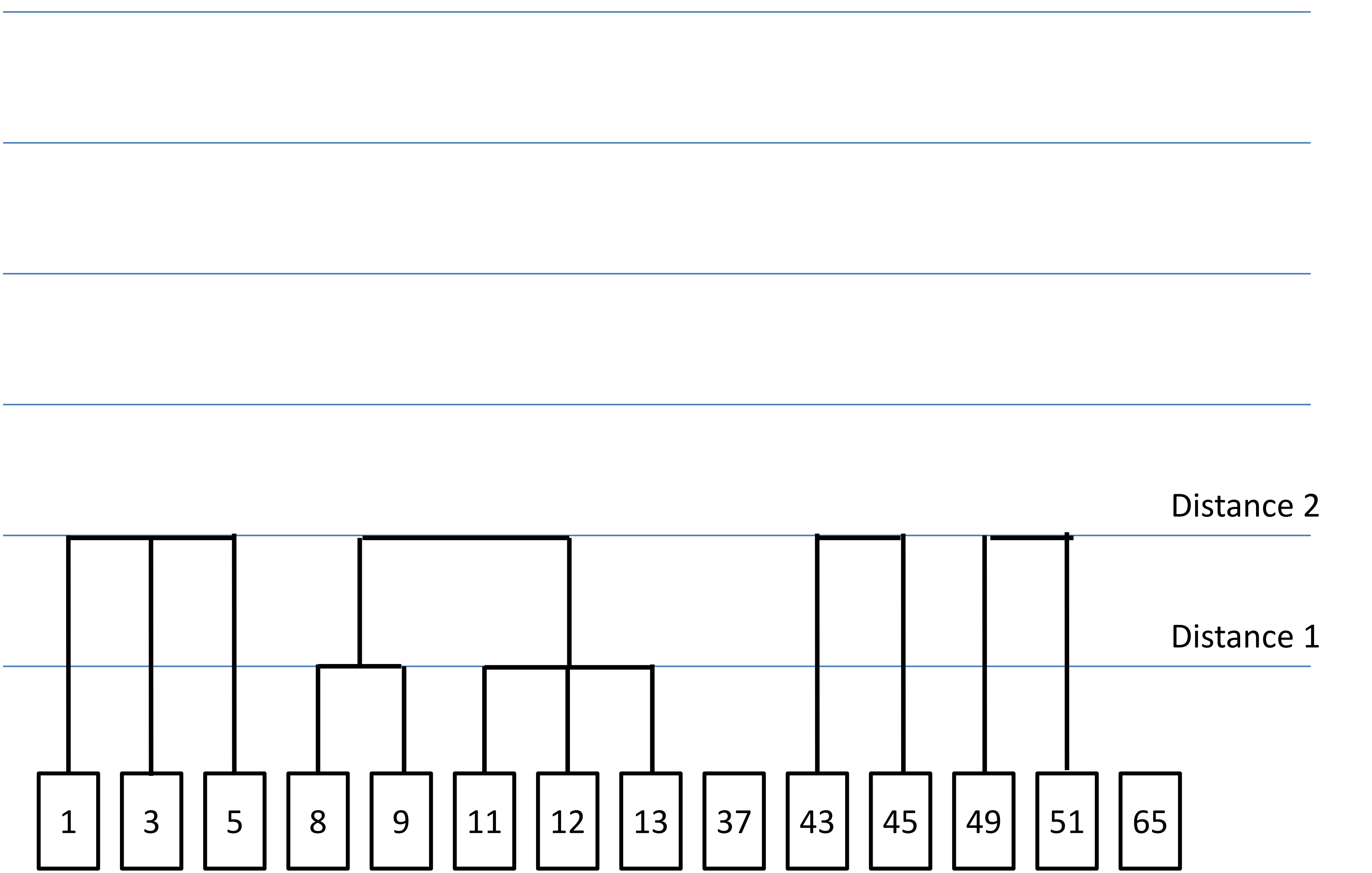
Example: clustering people by age

- Example in one dimension (to skip proximity matrix computation)
- The data consists of the ages of people at a family gathering.
- The goal is to cluster participants by age
- The distance between people is the difference in their ages.
- The procedure: sort participants by age, then begin clustering the closest groups

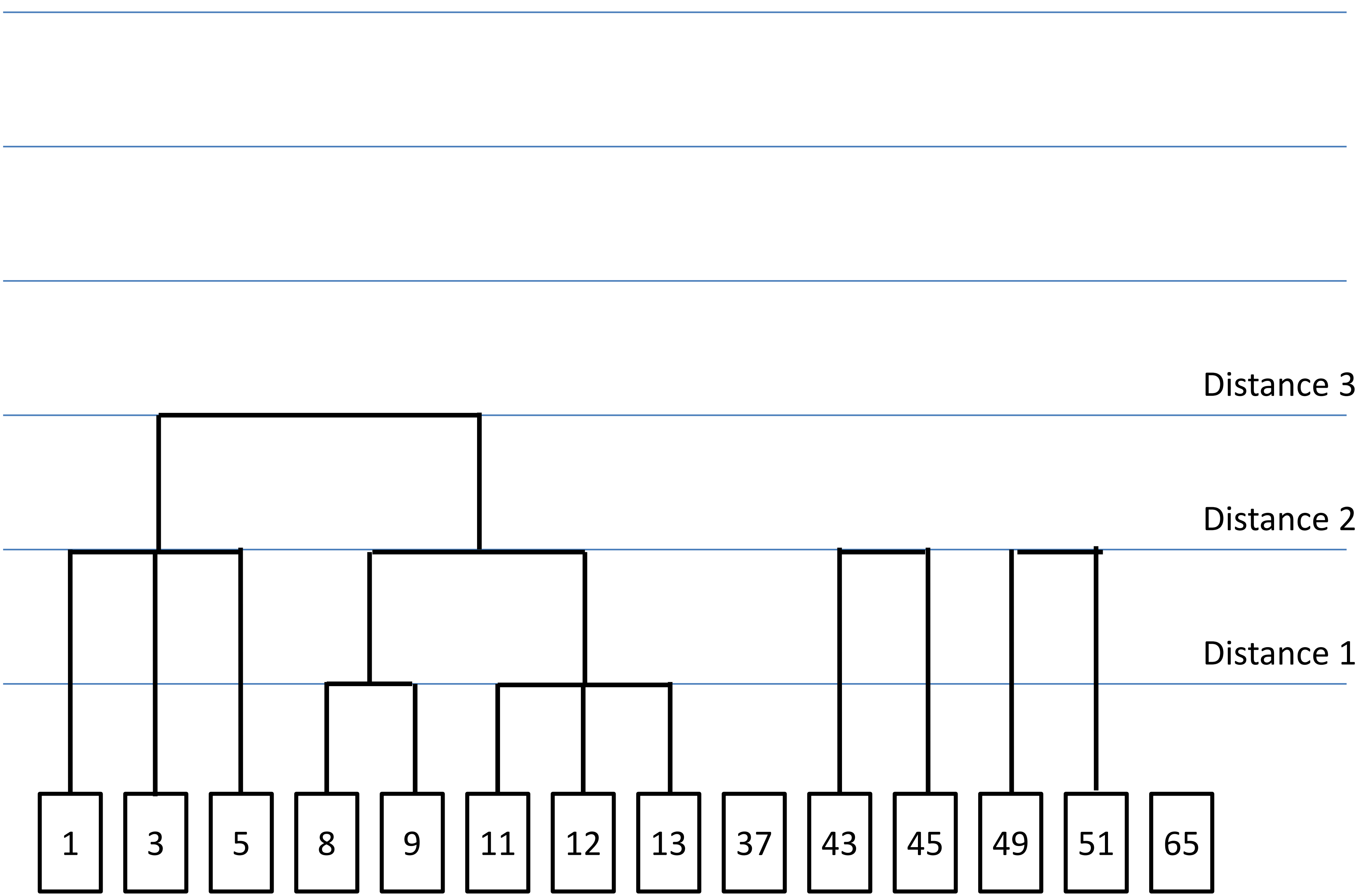
Distance between clusters: MIN



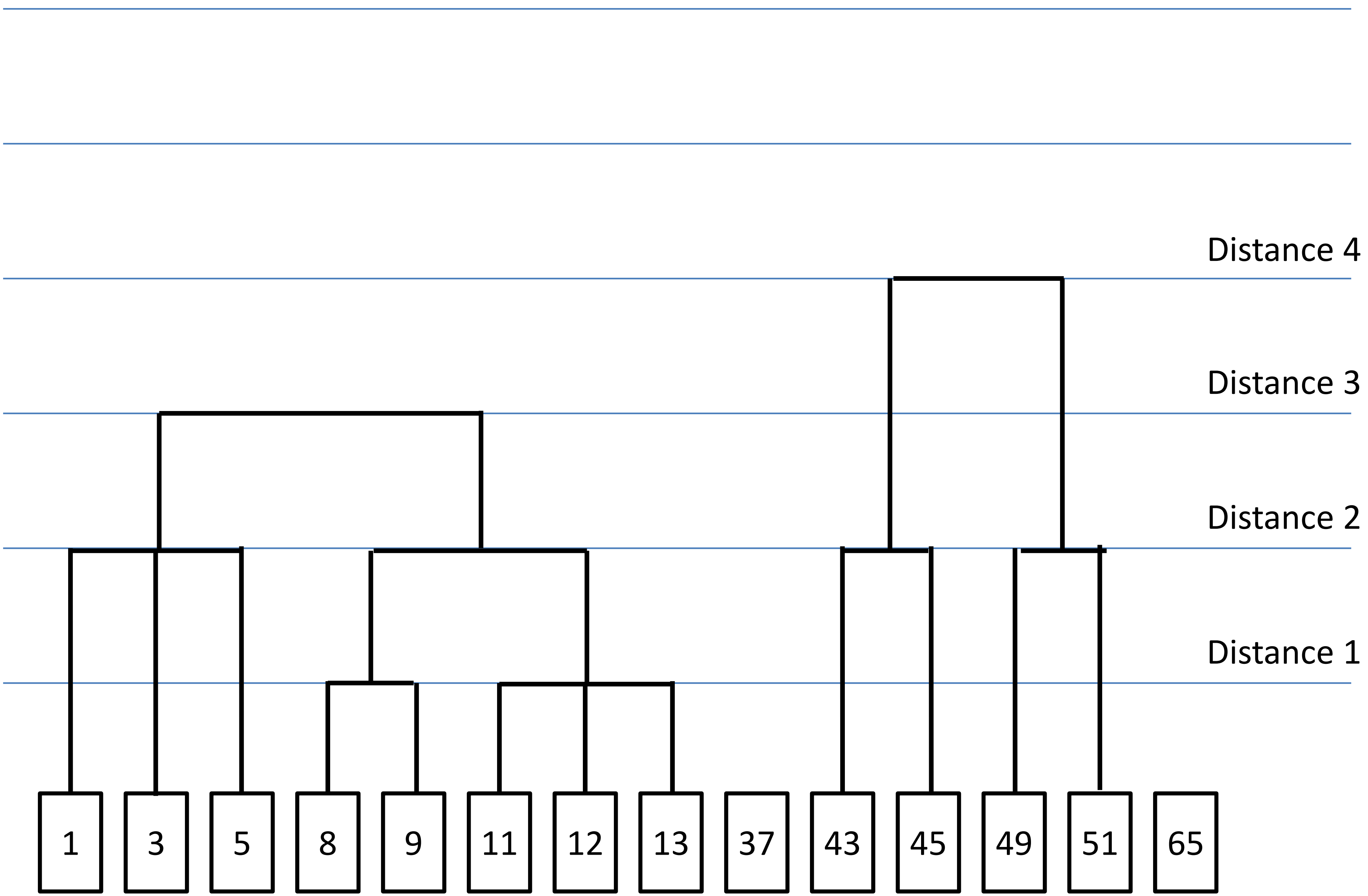
Distance between clusters: MIN



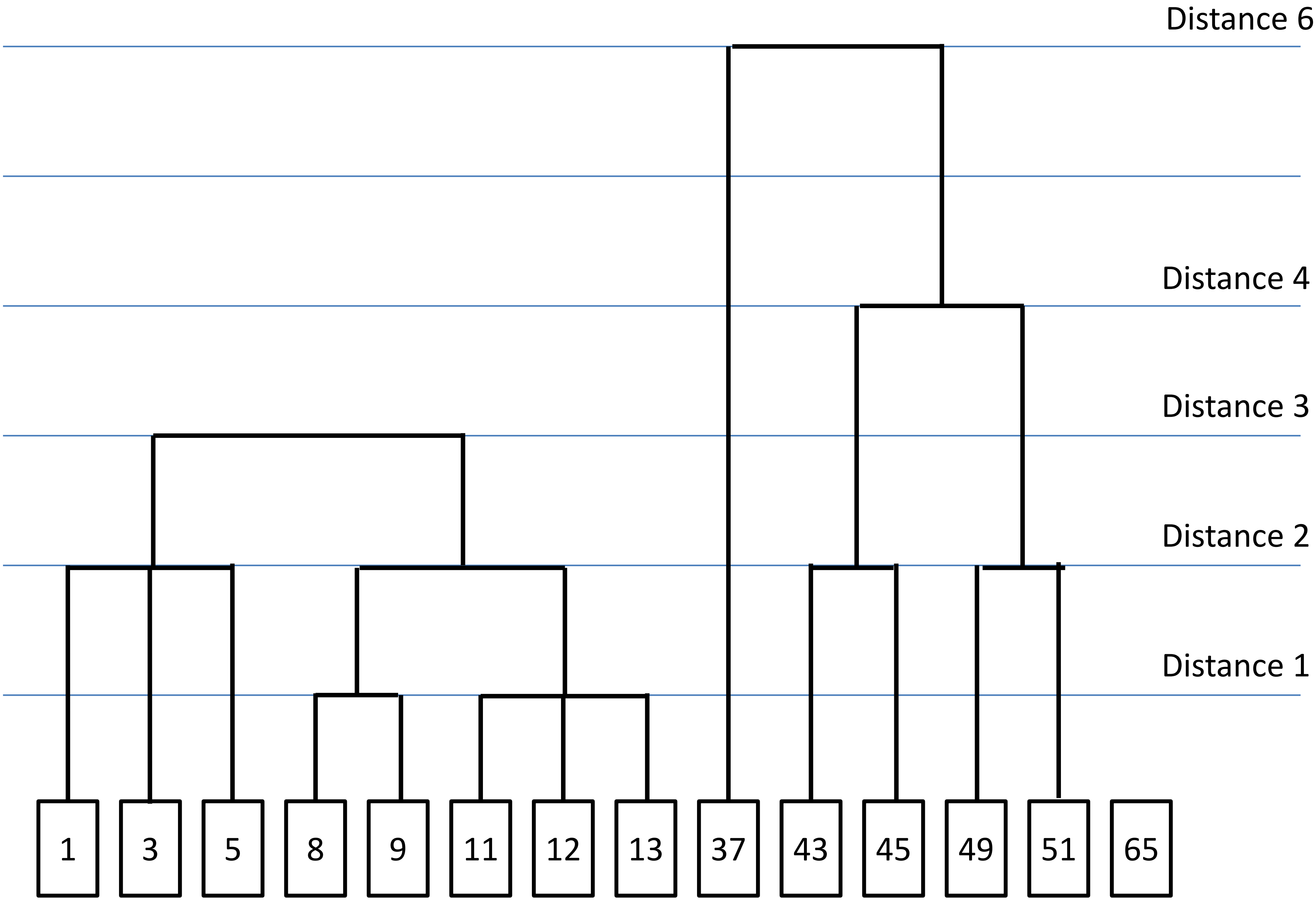
Distance between clusters: MIN



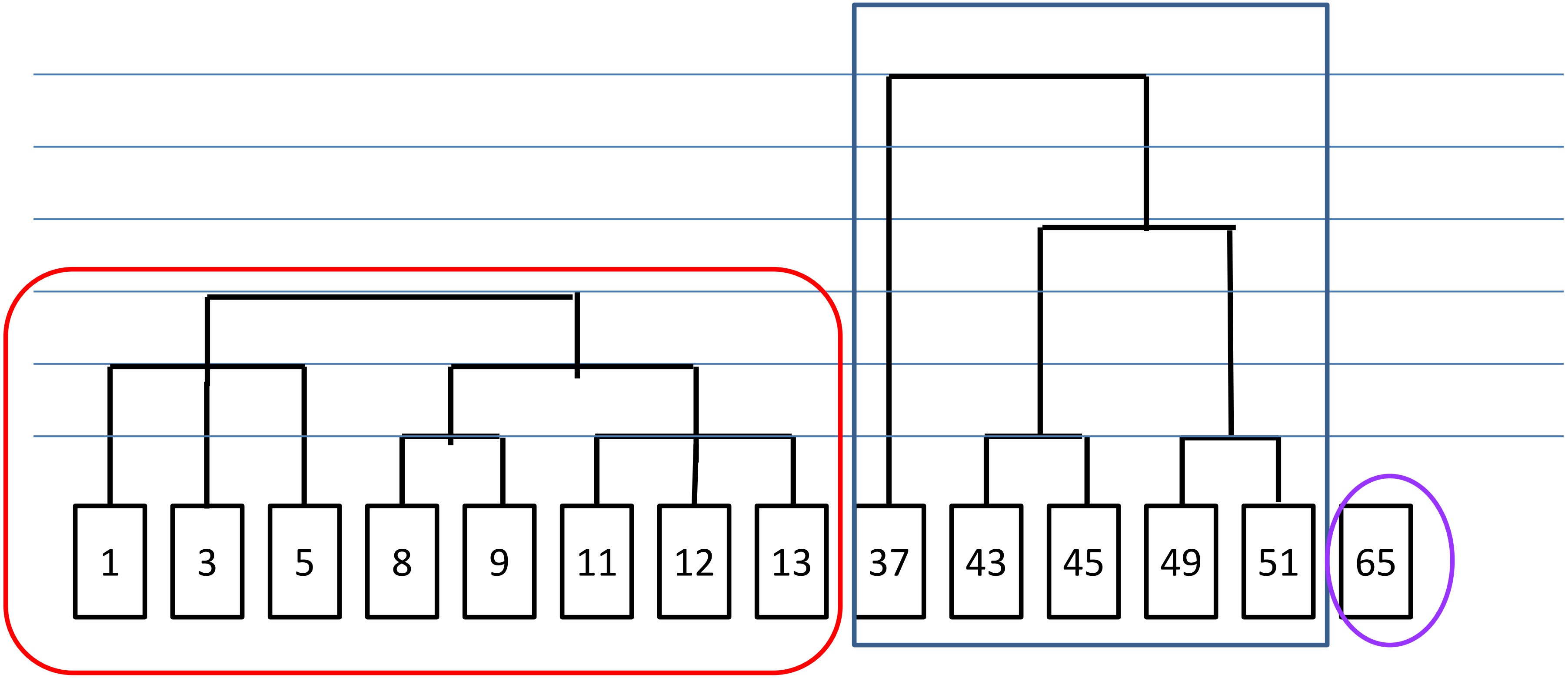
Distance between clusters: MIN



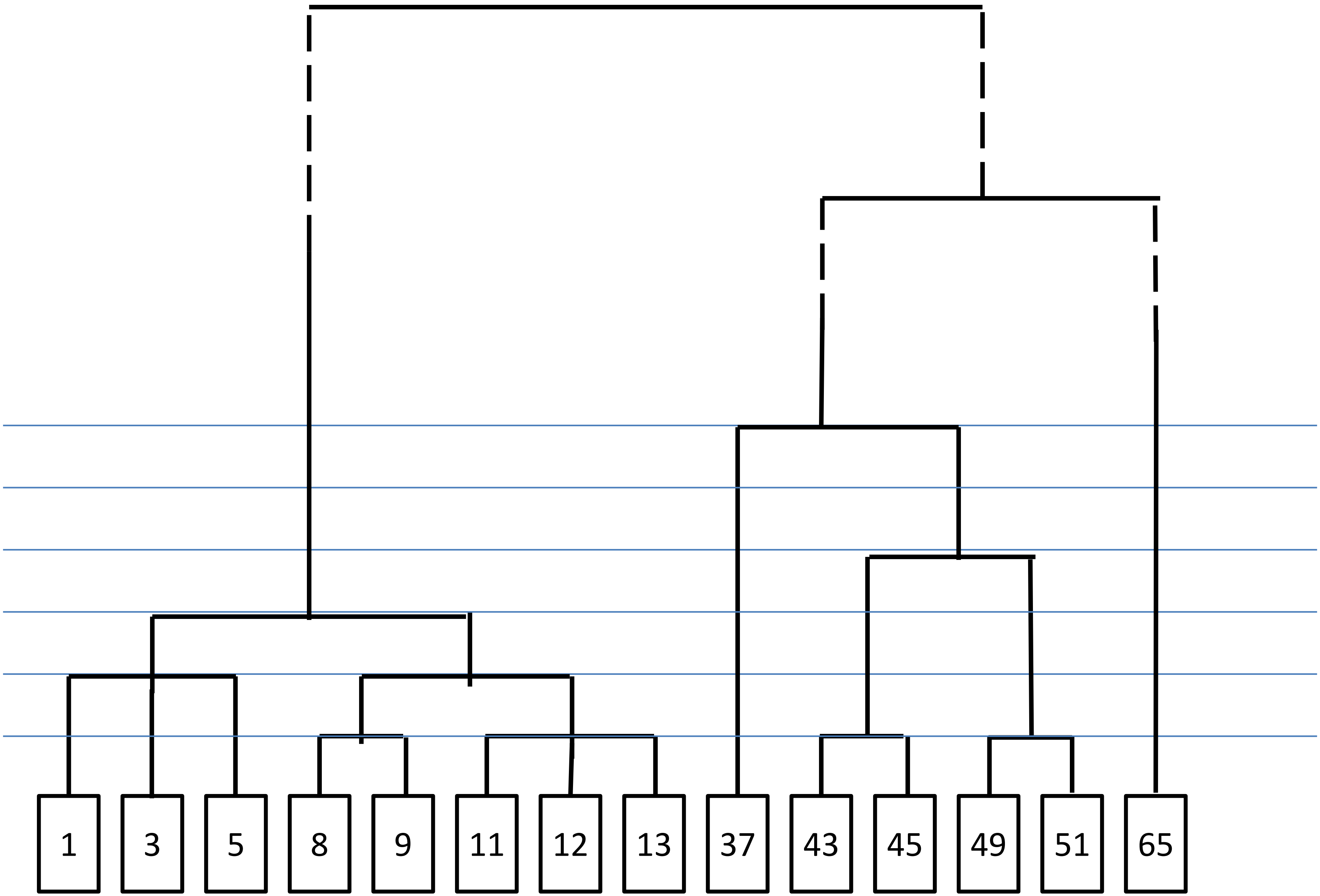
Distance between clusters: MIN



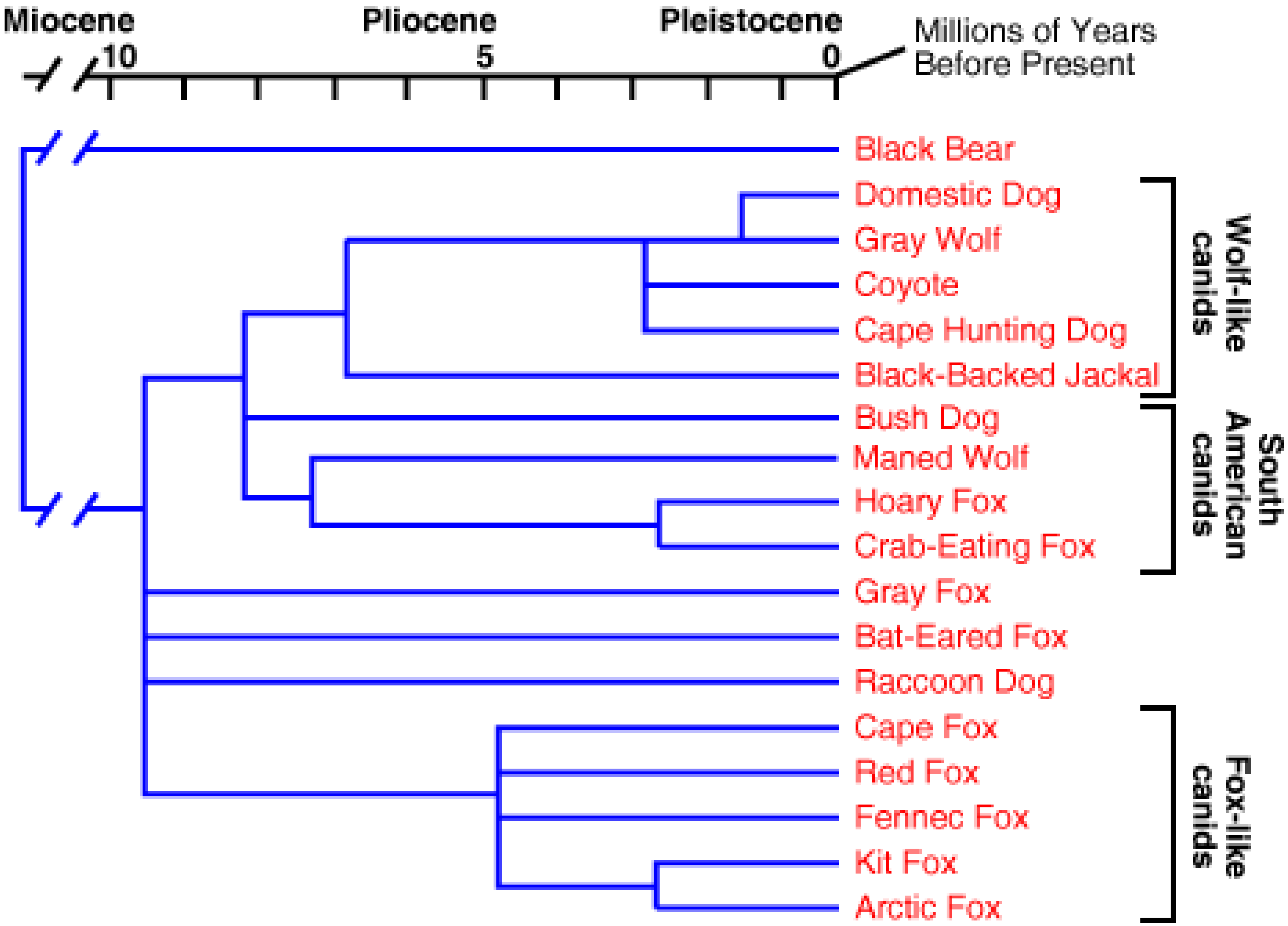
3 groups detected



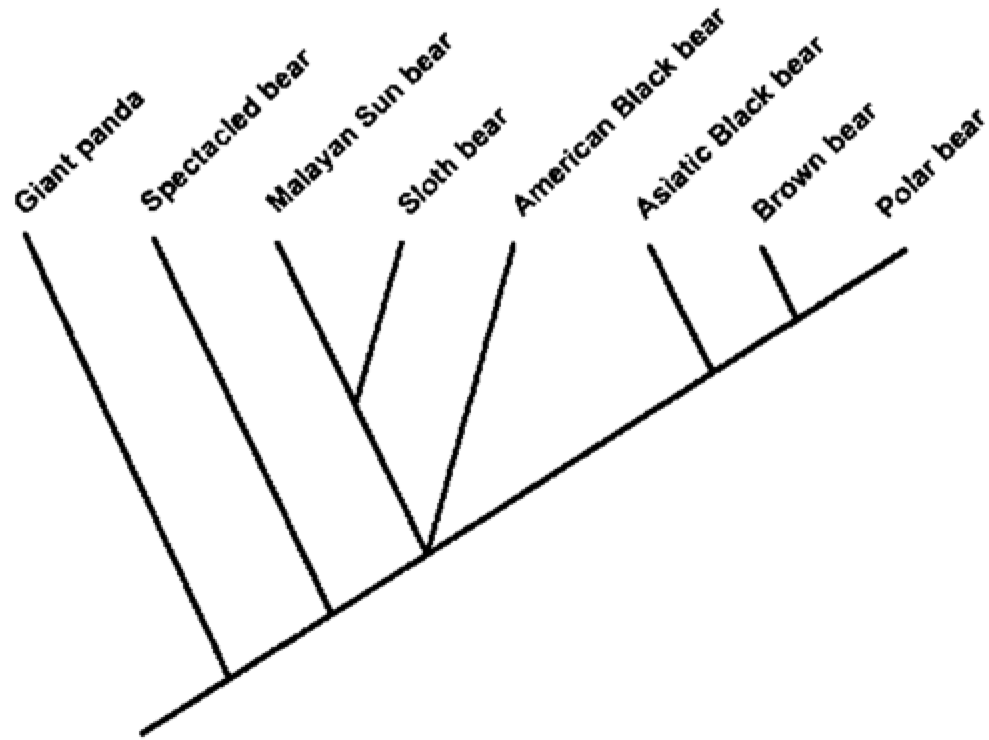
Final dendrogram



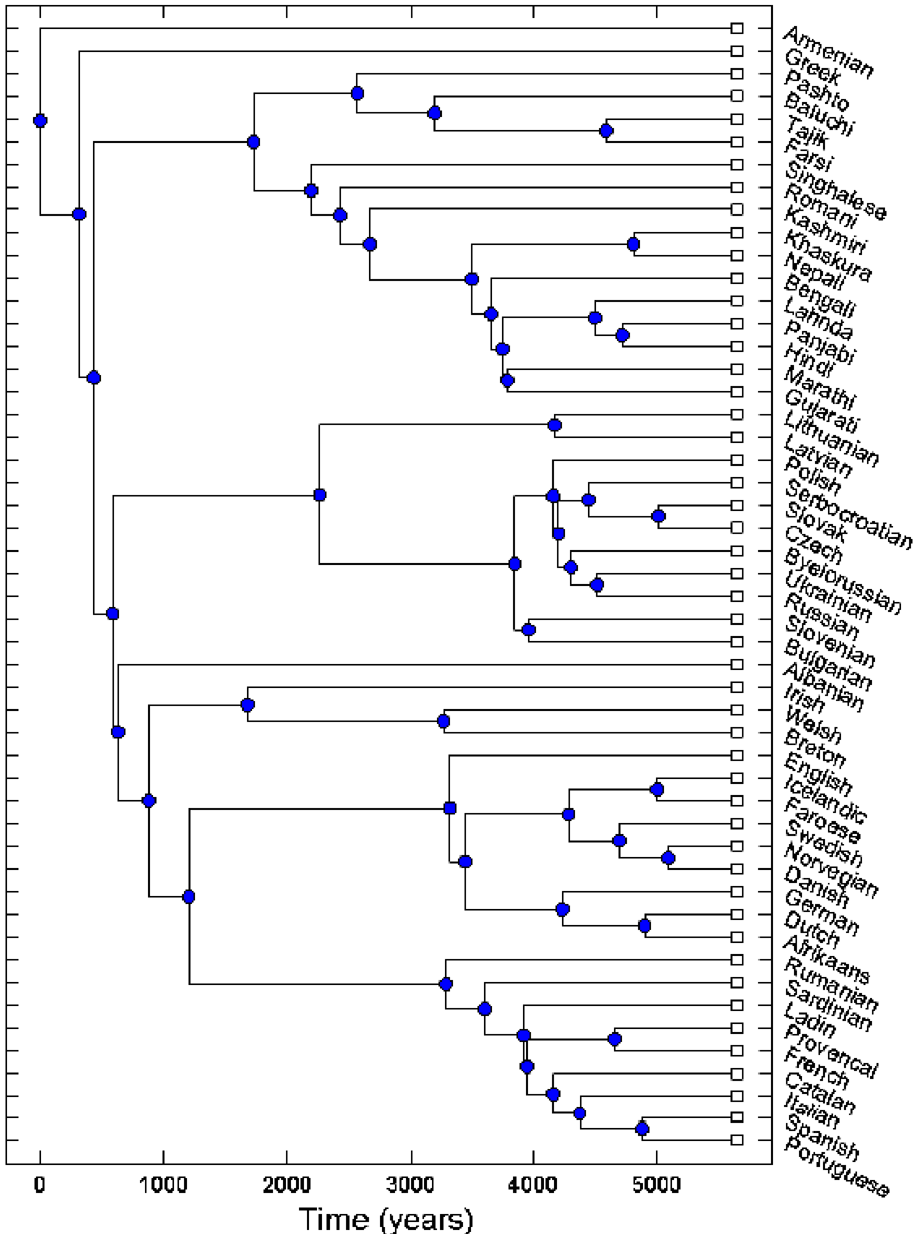
Hierarchical clustering application: evolution of Canidae



Giant Panda is a bear

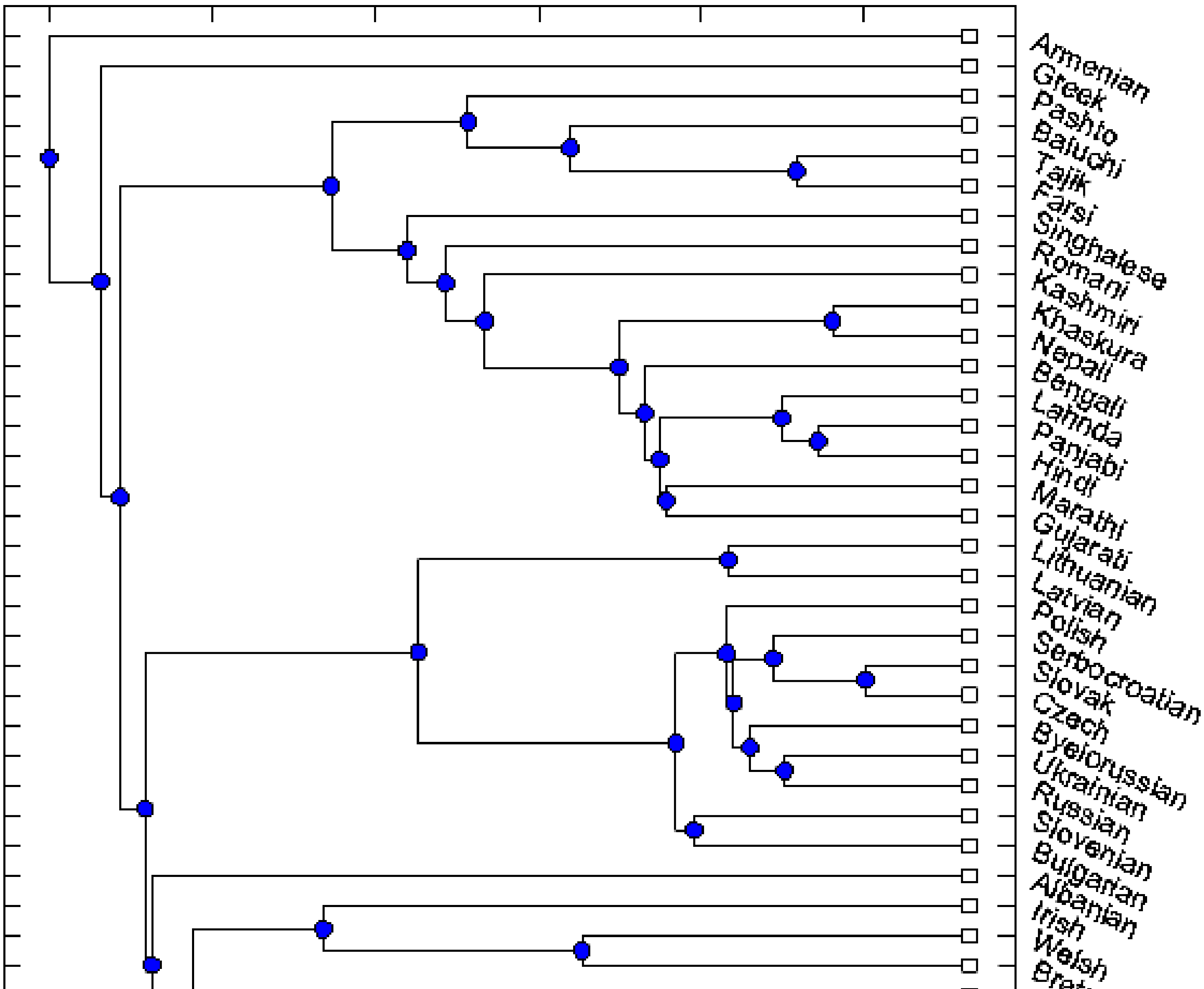


Hierarchical clustering application: languages evolution



From
"Indo-European languages tree
by Levenshtein distance"
by M. Serval and F. Petroni

Hierarchical clustering application: languages evolution

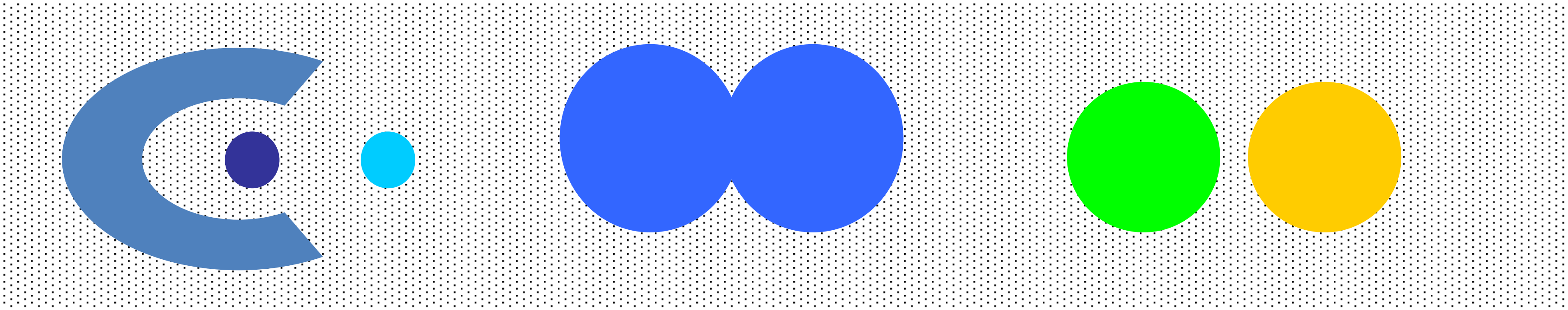


Clustering algorithms

- *K*-means clustering
- Agglomerative hierarchical clustering
- Density-based clustering

Types of Clusters: Density-Based

- Clusters are defined as dense regions of objects in the data space that are separated by regions of low density (representing noise)
- To discover such clusters we need special algorithms



6 density-based clusters

DBSCAN - Density-Based Spatial Clustering of Applications with Noise

New definitions

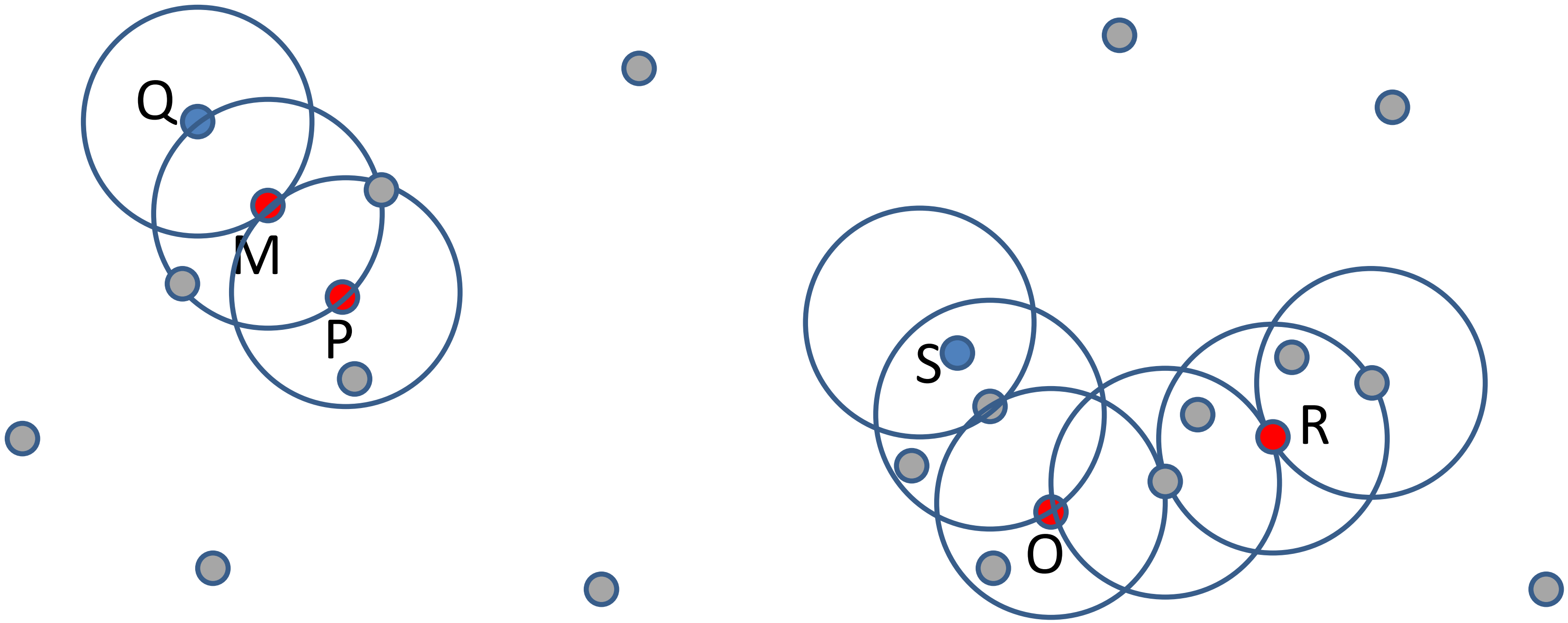
- The neighborhood within a radius ε of a given object is called the *ε-neighborhood* of the object
- If the ε-neighborhood of an object contains at least a minimum number *MinPts* of objects, then such an object is called **a core point**

DBSCAN - Density-Based Spatial Clustering of Applications with Noise

New definitions

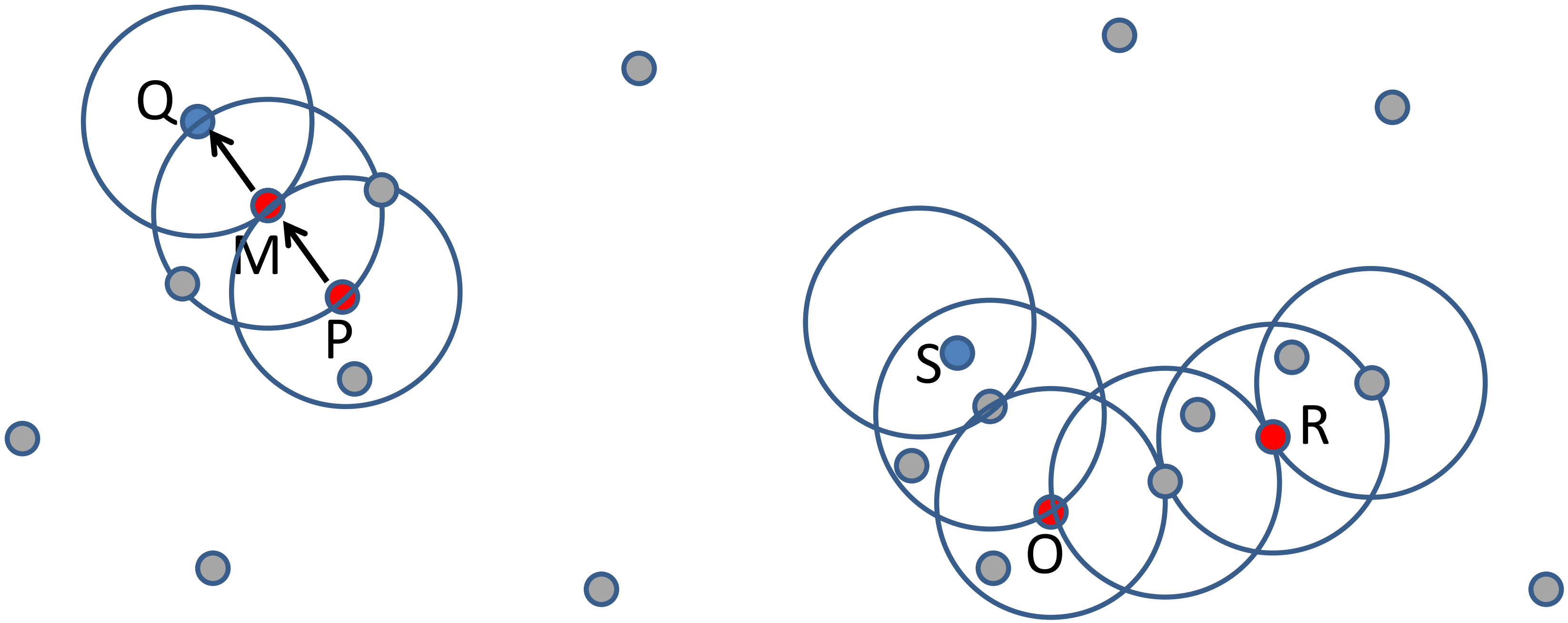
- We say that object *p* is directly reachable from object *q* if *p* is within ε-neighborhood of *q*, and *q* is a **core point**
- A **border point** has fewer than *MinPts* objects in its ε-neighborhood , but is **directly reachable from some core point**
- A **noise point** is any point that is neither a core point nor a border point.

Definitions: example: MinPts=3



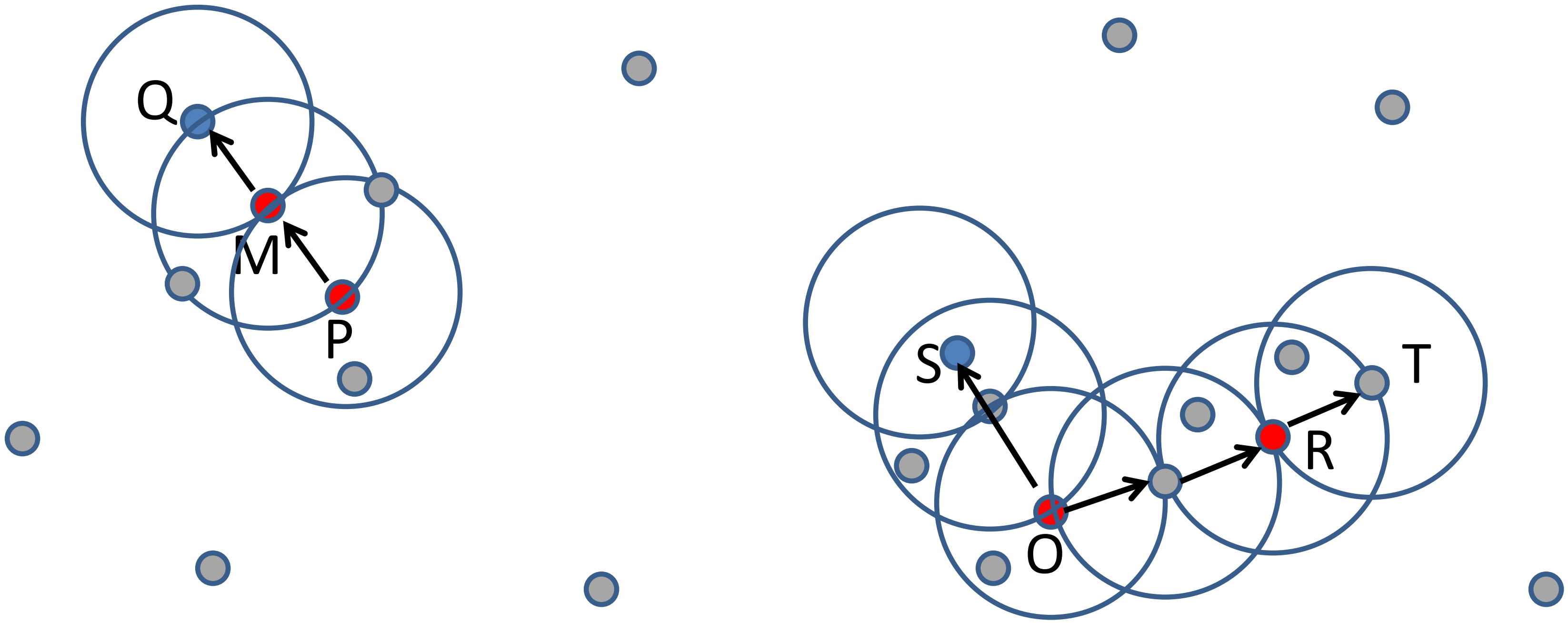
M, P, O and R are core points, since each contains at least 3 points in its ε-neighborhood

Definitions: example: MinPts=3



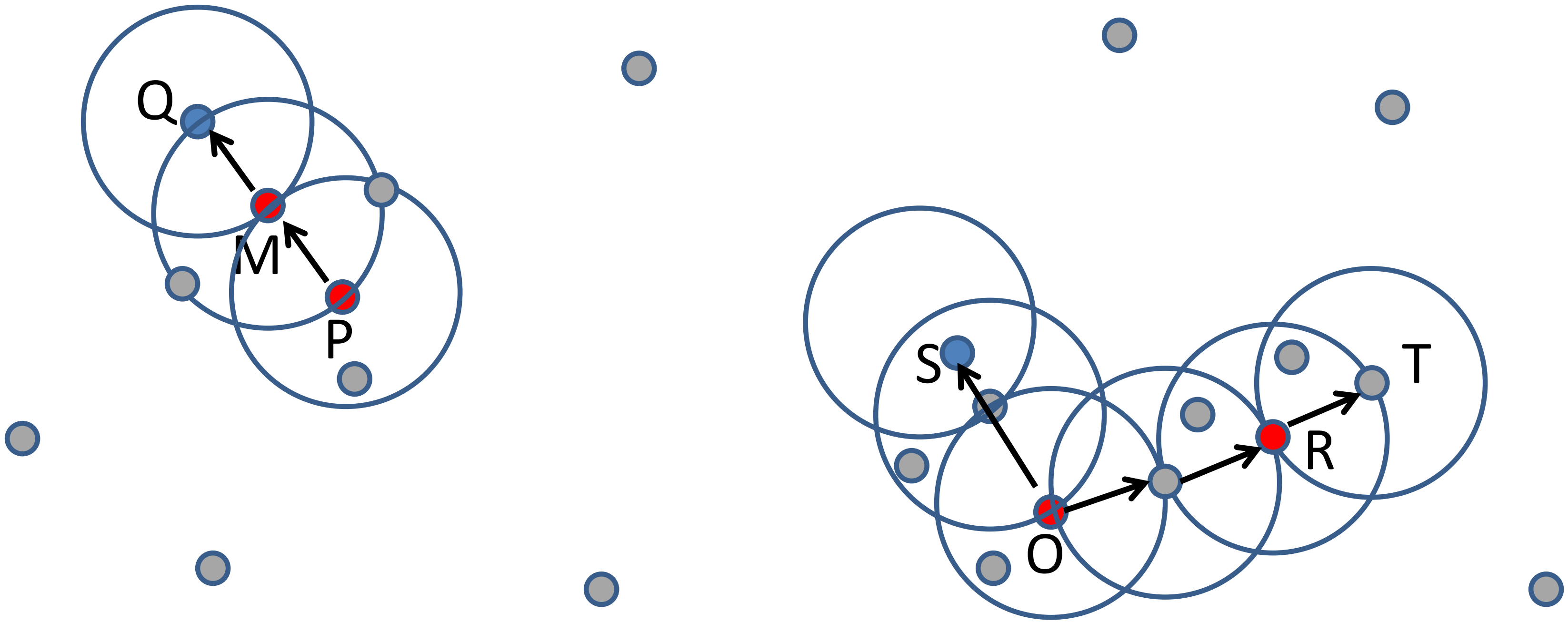
Q is directly density-reachable from M, M is directly density reachable from P, and P is directly density-reachable from M

Definitions: example: MinPts=3



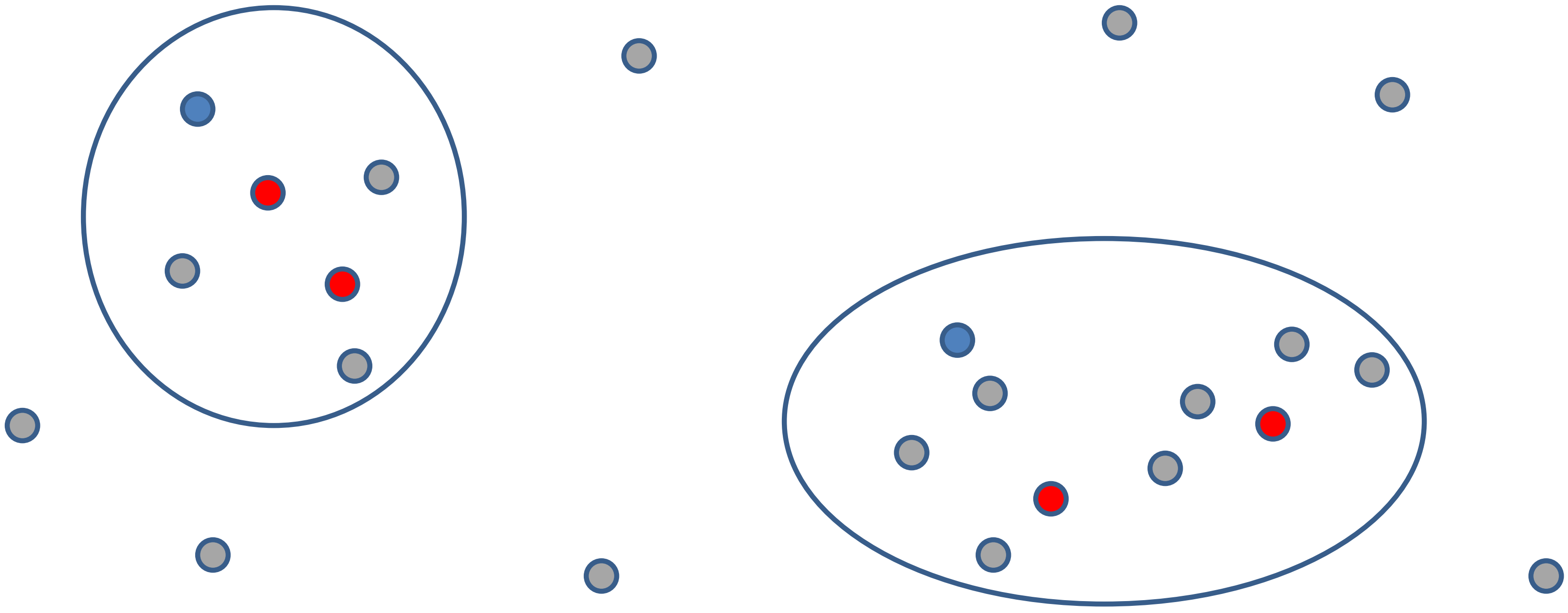
S is directly density-reachable from O, T is indirectly density-reachable from O, and T is directly density-reachable from R

Definitions: example: MinPts=3



O, R, S, T are density-connected

Density-based cluster



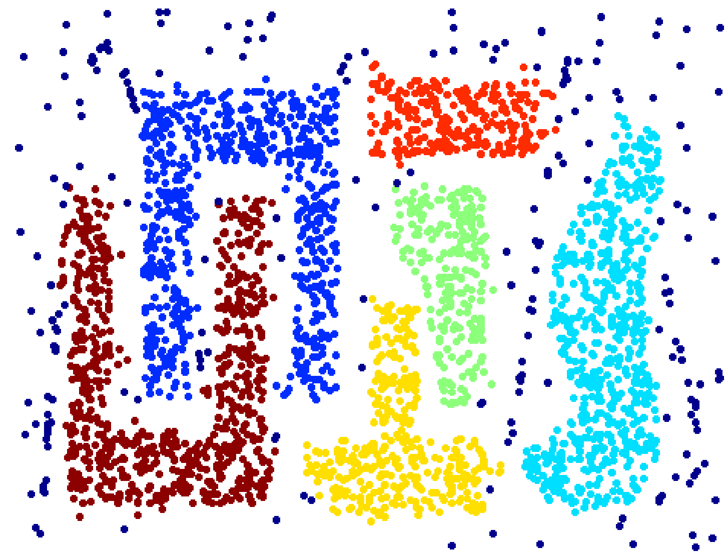
- A density-based cluster is a set of density-connected objects that is **maximal** with respects to density-reachability

DBSCAN algorithm

1. Check ε-neighborhood of each point and label each point as core, border, or noise point
2. Eliminate noise points
3. Combine all core points which are density-reachable into a single cluster
4. Assign each border point to one of the clusters of its associated core points

When DBSCAN Works Well

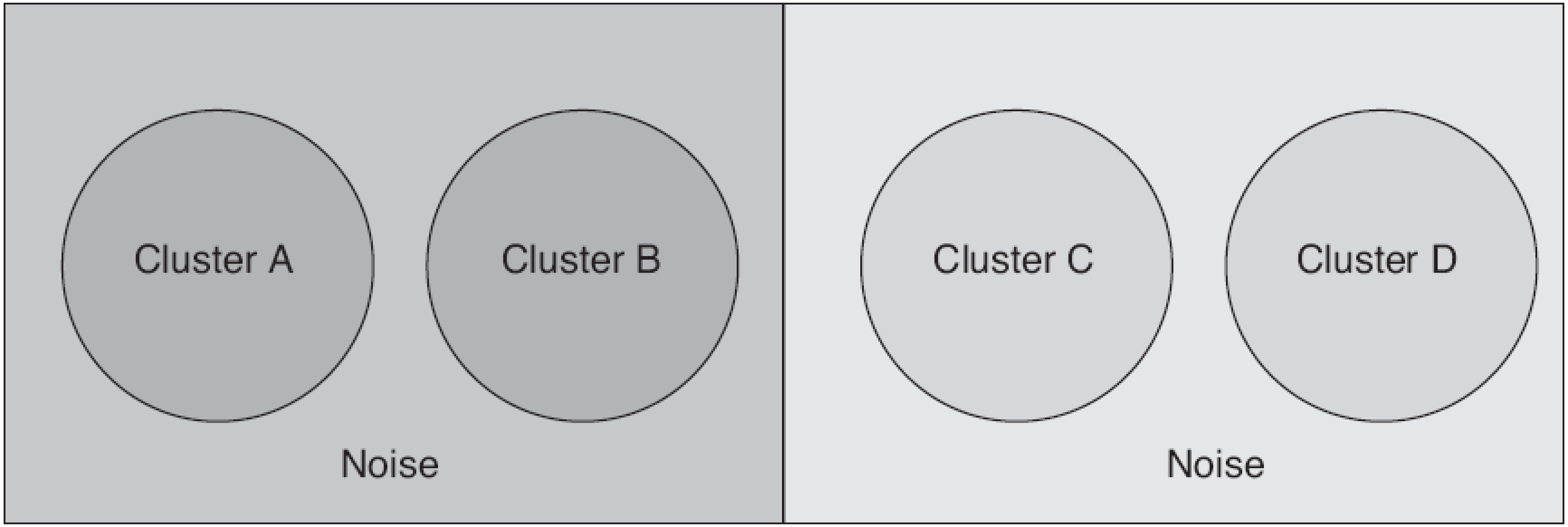
Original Points



Clusters

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

When DBSCAN Does NOT Work Well



Why DBSCAN doesn't work well here?

Selecting ε and MinPts

- If the radius is too large, than all points are core points
- If the radius is too small, then all points are outliers

Method for selecting DBSCAN parameters

- Decide how many points you want in a dense region: MinPts. Suppose we want core points to have at least *k* ε-neighbors
- Determine the distance from each point to its *k*-th nearest neighbor, called the *k*dist.
- For points that belong to some cluster, the value of *k*dist will be small [if *k* is not larger than the cluster size].
- However, for points that are not in a cluster, such as noise points, the *k*dist will be relatively large.

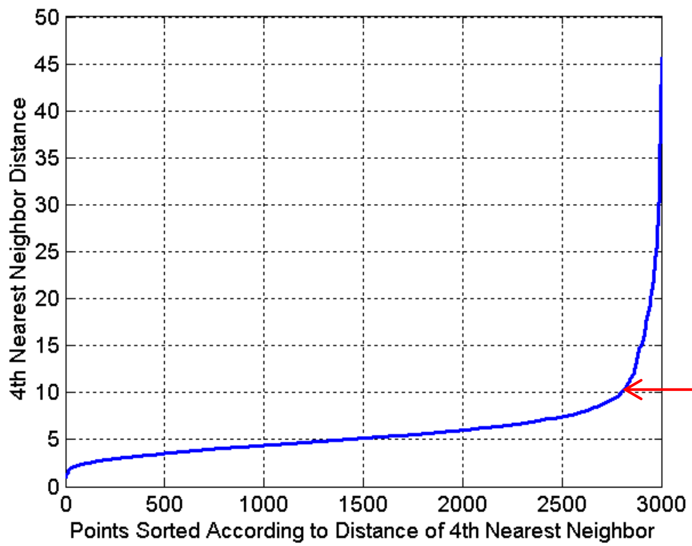
Method for selecting DBSCAN parameters

- So, if we compute the *k*dist for all the data points for some *k*, sort them in increasing order, and then plot the sorted values, we expect to see a **sharp change** at the value of *k*dist that corresponds to a suitable value of ε.

- If we select this dividing distance as the ε parameter and take the value of *k* as the MinPts parameter, then points for which *k*dist is less than ε will be labeled as core points, while other points will be labeled as noise or border points.

- If there is no sharp change in distance then
  - the entire dataset is a noise, or
  - change value of *k*

DBSCAN: Determining EPS and MinPts



Use distance 10 to separate clusters from noise

- E determined in this way depends on *k*, but does not change dramatically as *k* changes.
- If *k* is too small ?

  then even a small number of closely spaced points that are noise or outliers will be incorrectly labeled as clusters.
- If *k* is too large ?

  then small clusters (of size less than *k*) are likely to be labeled as noise.
- Original DBSCAN used *k* = 4, which appears to be a reasonable value for most data sets.