

Hierarchical Methods (BIRCH)

Agglomerative Algorithm

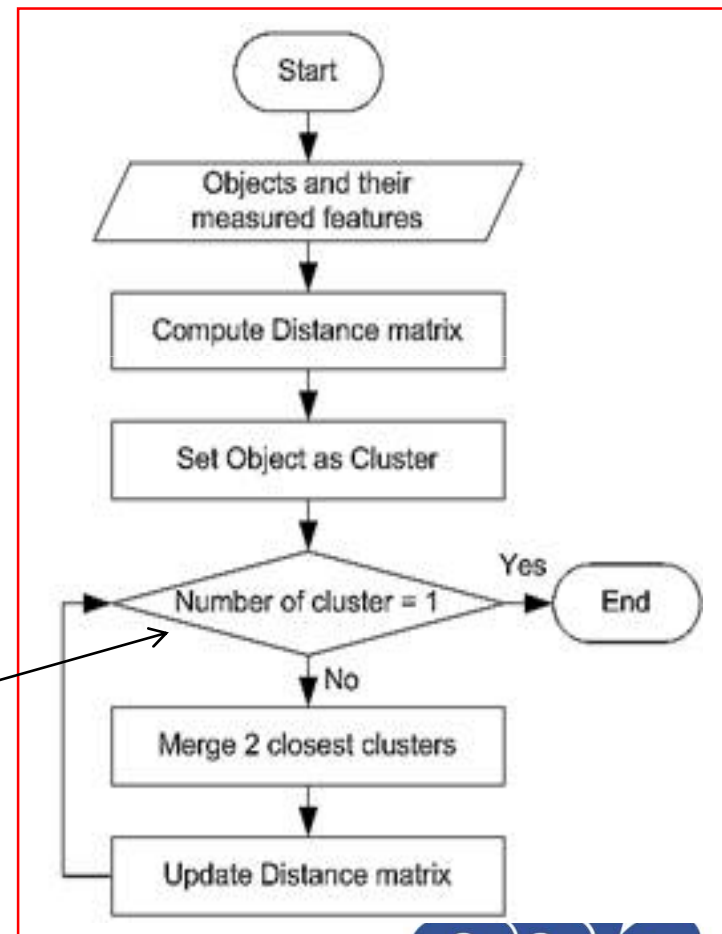
- The Agglomerative algorithm is carried out in three steps:

1) Convert all object features into a distance matrix

2) Set each object as a cluster (thus if we have N objects, we will have N clusters at the beginning)

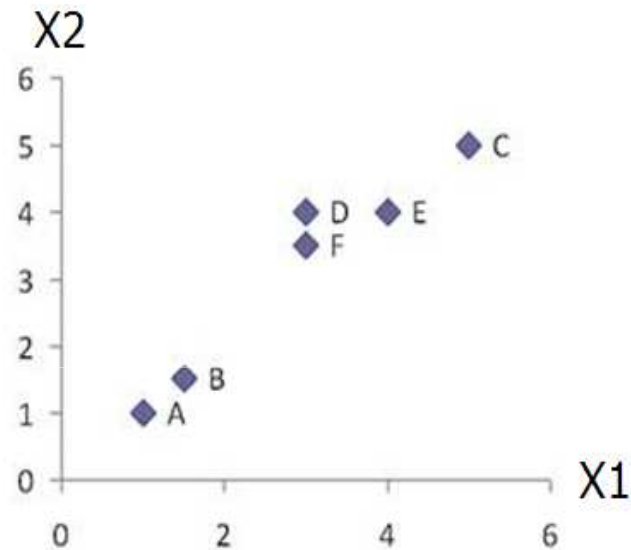
3) Repeat until number of cluster is one (or known # of clusters)

- Merge two closest clusters
- Update "distance matrix"



Example

- Problem: clustering analysis with agglomerative algorithm



	X1	X2
A	1	1
B	1.5	1.5
C	5	5
D	3	4
E	4	4
F	3	3.5

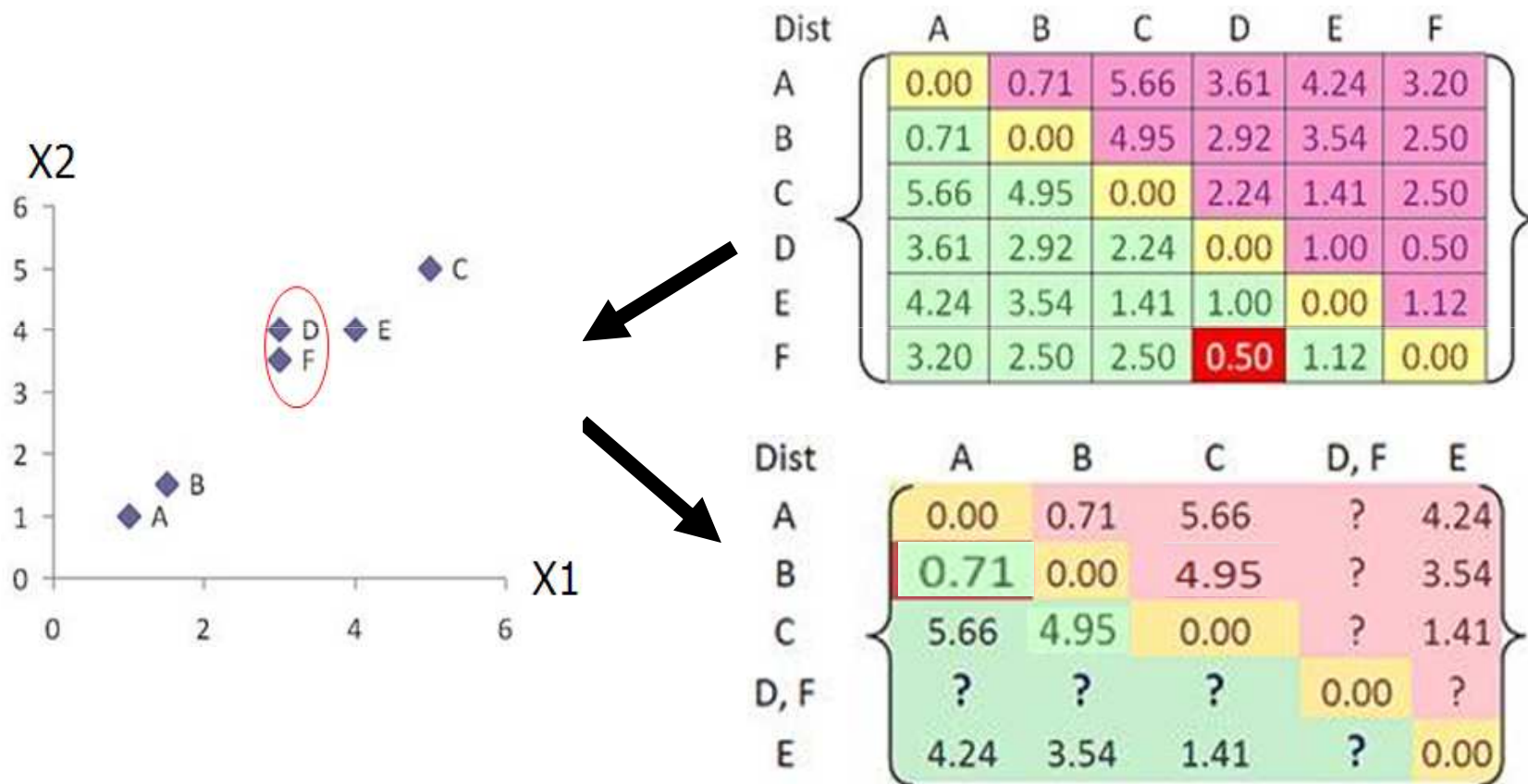
$$d_{AB} = \left((1-1.5)^2 + (1-1.5)^2 \right)^{\frac{1}{2}} = \sqrt{\frac{1}{2}} = 0.7071$$

$$d_{DF} = \left((3-3)^2 + (4-3.5)^2 \right)^{\frac{1}{2}} = 0.5$$

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

Example

- Merge two closest clusters (iteration 1)



Example

- Update distance matrix (iteration 1)

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

$$d_{(D,F) \rightarrow A} = \min(d_{DA}, d_{FA}) = \min(3.61, 3.20) = 3.20$$

$$d_{(D,F) \rightarrow B} = \min(d_{DB}, d_{FB}) = \min(2.92, 2.50) = 2.50$$

$$d_{(D,F) \rightarrow C} = \min(d_{DC}, d_{FC}) = \min(2.24, 2.50) = 2.24$$

$$d_{E \rightarrow (D,F)} = \min(d_{ED}, d_{EF}) = \min(1.00, 1.12) = 1.00$$

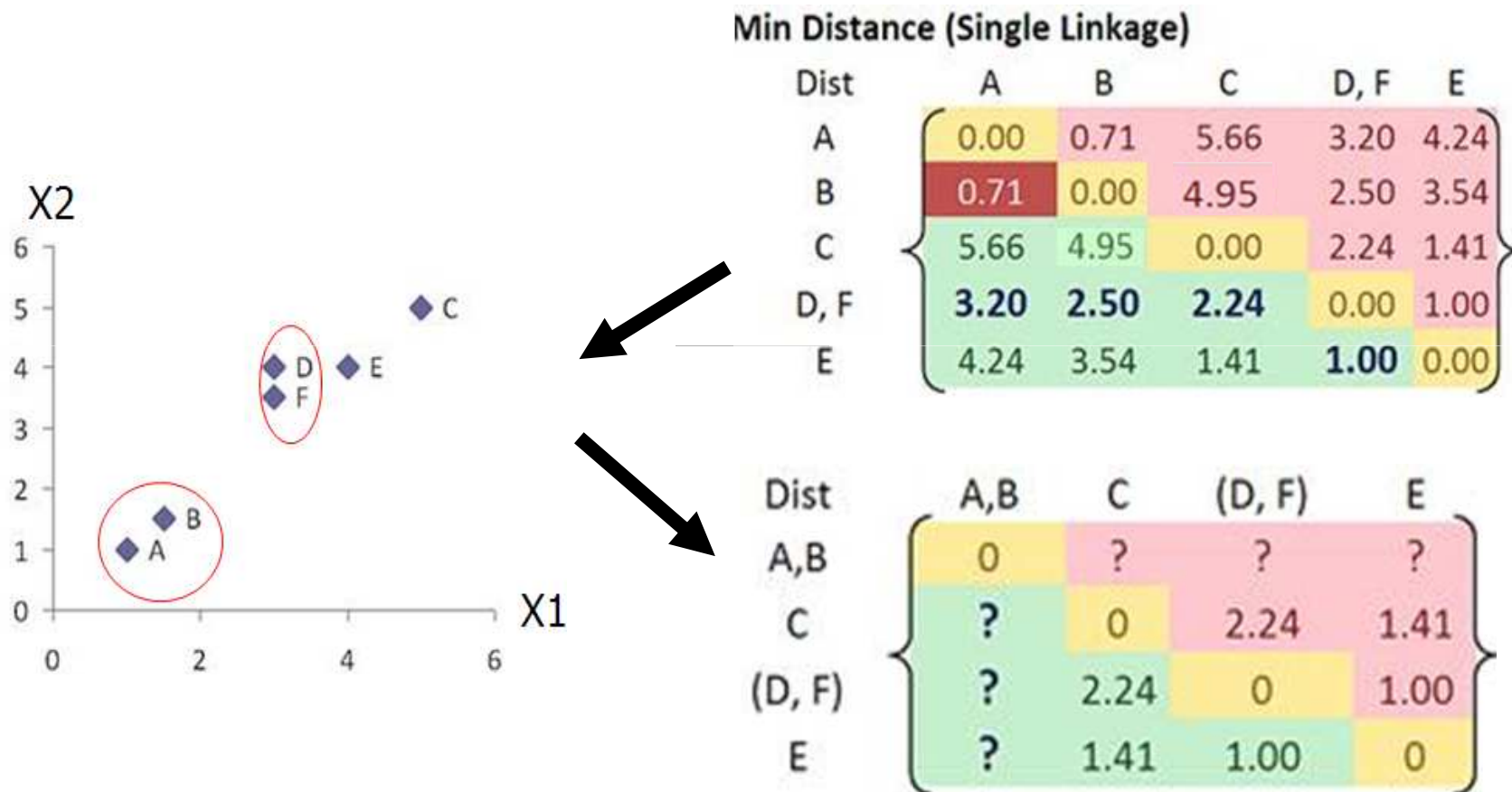
Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	3.20	2.50	2.24	0.00	1.00
E	4.24	3.54	1.41	1.00	0.00

Example

- Merge two closest clusters (iteration 2)



Example

- Update distance matrix (iteration 2)

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	3.20	2.50	2.24	0.00	1.00
E	4.24	3.54	1.41	1.00	0.00

$$d_{C \rightarrow (A,B)} = \min(d_{CA}, d_{CB}) = \min(5.66, 4.95) = 4.95$$

$$d_{(D,F) \rightarrow (A,B)} = \min(d_{DA}, d_{DB}, d_{FA}, d_{FB}) \\ = \min(3.61, 2.92, 3.20, 2.50) = 2.50$$

$$d_{E \rightarrow (A,B)} = \min(d_{EA}, d_{EB}) = \min(4.24, 3.54) = 3.54$$

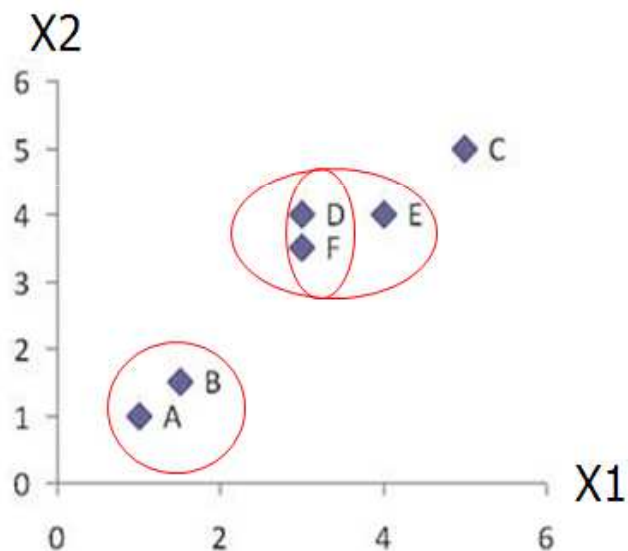
Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	?	0	2.24	1.41
(D, F)	?	2.24	0	1.00
E	?	1.41	1.00	0

Min Distance (Single Linkage)

Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	4.95	0	2.24	1.41
(D, F)	2.50	2.24	0	1.00
E	3.54	1.41	1.00	0

Example

- Merge two closest clusters/update distance matrix (iteration 3)



Min Distance (Single Linkage)

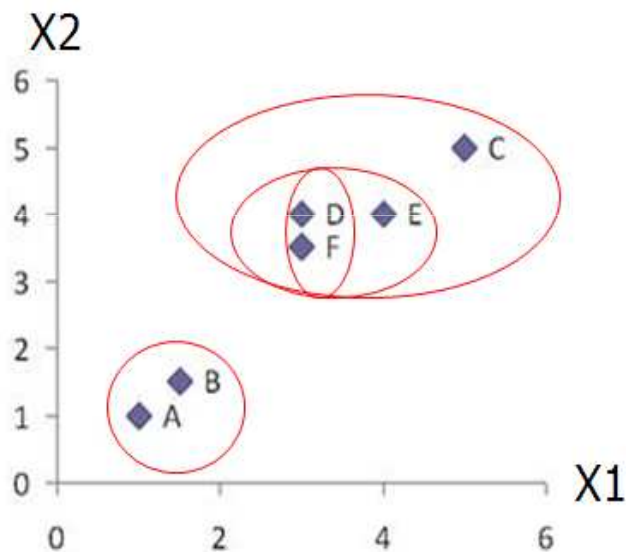
Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	4.95	0	2.24	1.41
(D, F)	2.50	2.24	0	1.00
E	3.54	1.41	1.00	0

Min Distance (Single Linkage)

Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	2.50	1.41	0.00

Example

- Merge two closest clusters/update distance matrix (iteration 4)



Min Distance (Single Linkage)

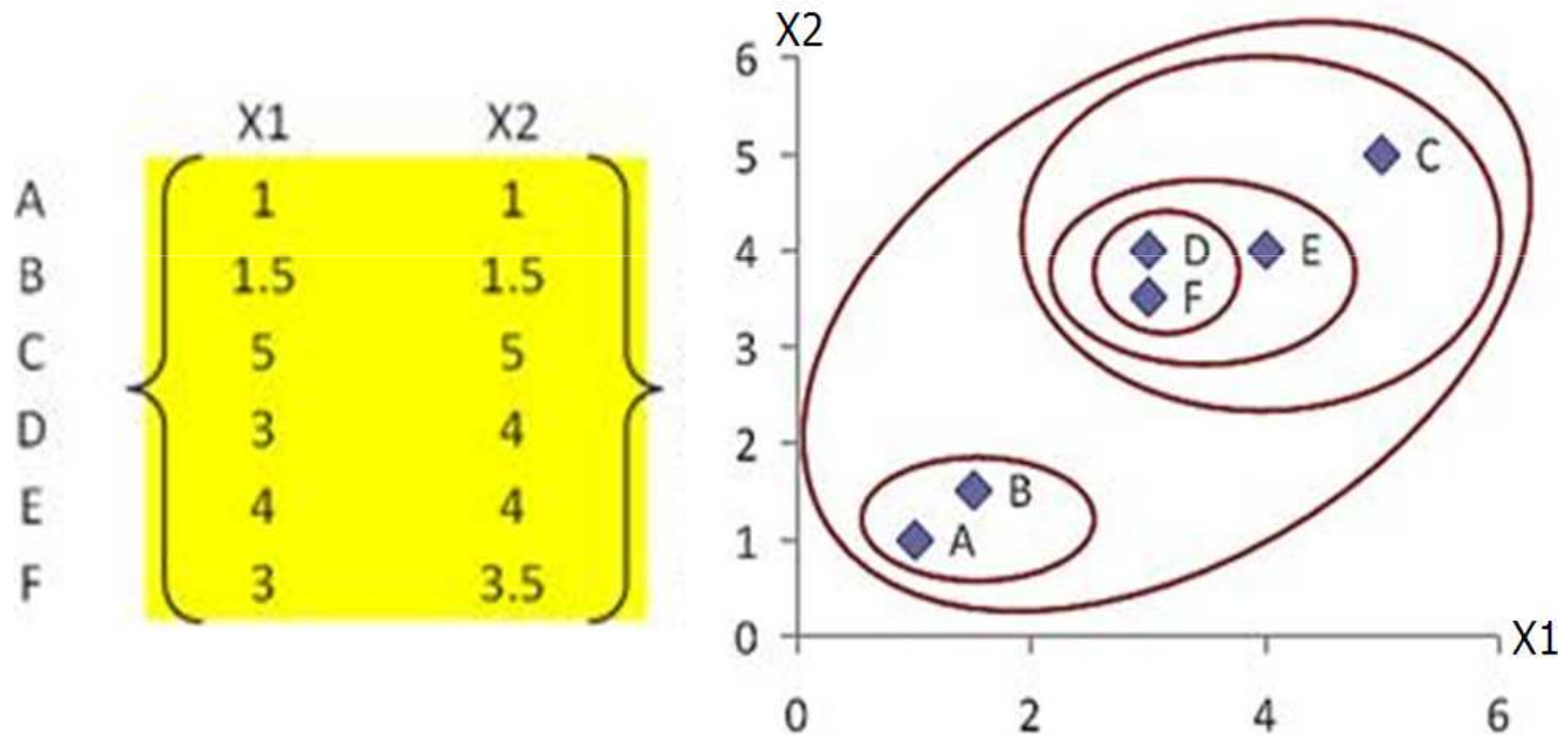
Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	2.50	1.41	0.00

Min Distance (Single Linkage)

Dist	(A,B)	((D, F), E), C
(A,B)	0.00	2.50
((D, F), E), C	2.50	0.00

Example

- Final result (meeting termination condition)



DIVISIVE METHOD

- Compared to agglomerative hierarchical clustering, divisive clustering proceeds in the opposite way.
- In the beginning, the entire data set belongs to a cluster, and a procedure successively divides it until all clusters are singletons.
- For a data set with N objects, a divisive hierarchical algorithm would start by considering $2^{N-1} - 1$ possible divisions of the data into two nonempty subsets, which is computationally expensive even for small-scale data sets.
- Therefore, divisive clustering is not a common choice in practice



DIANA


- DIANA (Divisive Analysis) considers only a part of all the possible divisions
- DIANA consists of a series of iterative steps in order to move the objects into the splinter group, which is seeded with the object that is farthest from the others in the cluster to be divided.
- The cluster with the largest diameter, defined as the largest distance between any pair of objects, is selected for further division

DIANA

- Suppose that cluster C_l is going to be split into clusters C_i and C_j :
 - Start with $C_i = C_l$ and C_j empty
 - For each data object x_m in C_i :
 - For the first iteration, compute its average distance to all the other objects:

$$d(x_m, C_i \setminus \{x_m\}) = \frac{1}{N_{C_i} - 1} \sum_{\substack{x_p \in C_i \\ p \neq m}} d(x_m, x_p)$$

- For the remaining iterations, compute the difference between the average distance to C_i and the average distance to C_j :

$$d(x_m, C_i \setminus \{x_m\}) - d(x_m, C_j) = \frac{1}{N_{C_i} - 1} \sum_{\substack{x_p \in C_i \\ p \neq m}} d(x_m, x_p) - \frac{1}{N_{C_j} - 1} \sum_{x_q \in C_j} d(x_m, x_q)$$


DIANA

- Decide whether to move element x_m to cluster C_j or keep it in cluster C_i :
 - For the first iteration, move the object with the maximum value to C_j (that is, the element farther from every other element is separated from the others)
 - For the remaining iterations, if the maximum difference value is greater than zero, move the data object with the maximum difference into C_j , then repeat steps 2b and 3b. If the maximum value is less than zero, stop.

Recent Advances

- Criticisms of classical hierarchical clustering algorithms:
 - Lack of robustness (sensitivity to noise and outliers)
 - Not capable of correcting possible previous misclassification
 - Computational complexity, which is at least $O(N^2)$

Recent Advances

- New clustering methods, with hierarchical cluster results, have appeared and have greatly improved the clustering performance.
 - BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)
 - Chameleon: Multiphase Hierarchical clustering with dynamic modeling
 - Probabilistic Hierarchical clustering

BIRCH

- *BIRCH is designed for clustering a large amount of numerical data*
- *It overcomes the two difficulties of agglomerative clustering methods:*
 - *scalability*
 - *the inability to undo what was done in the previous step.*

BIRCH

- BIRCH introduces two concepts:
 - Clustering Feature (CF)
 - Clustering feature tree (CF tree)
- They are used to summarize cluster representations.
- These structures help the clustering method achieve good speed and scalability in large databases and also make it effective for incremental and dynamic clustering of incoming objects.

BIRCH Algorithm

- Given n d -dimensional data objects or points in a cluster, we can define the centroid \mathbf{x}_0 , radius \mathbf{R} , and diameter \mathbf{D} of the cluster as follows:

$$\mathbf{x}_0 = \frac{\sum_{i=1}^n \mathbf{x}_i}{n} \quad \mathbf{R} = \sqrt{\frac{\sum_{i=1}^n (\mathbf{x}_i - \mathbf{x}_0)^2}{n}} \quad \mathbf{D} = \sqrt{\frac{\sum_{i=1}^n \sum_{j=1}^n (\mathbf{x}_i - \mathbf{x}_j)^2}{n(n-1)}}$$

- Where \mathbf{R} is the average distance from member objects to the centroid, and \mathbf{D} is the average pairwise distance within a cluster.
- Both \mathbf{R} and \mathbf{D} reflect the tightness of the cluster around the centroid.

BIRCH Algorithm

- **Clustering Feature (CF)**

- CF is a three-dimensional vector summarizing information about clusters of objects.
- Given n d -dimensional objects or points in a cluster, $\{x_i\}$, then the CF of the cluster is defined as:

$$CF = \langle n, LS, SS \rangle$$

- where n is the number of points in the cluster,
- LS is the linear sum of the n points, i.e.,

$$\sum_{i=1}^n x_i$$

- SS is the square sum of the data points, i.e.,

$$\sum_{i=1}^n x_i^2$$

BIRCH Algorithm

- Clustering features are **additive**.
- For example, suppose that we have two disjoint clusters, C_1 and C_2 , having the clustering features, CF_1 and CF_2 , respectively.
- The clustering feature for the cluster that is formed by merging C_1 and C_2 is simply $CF_1 + CF_2$.
- Clustering features are sufficient for calculating all of the measurements that are needed for making clustering decisions in BIRCH.

BIRCH Algorithm

- **Example: Clustering feature.**

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

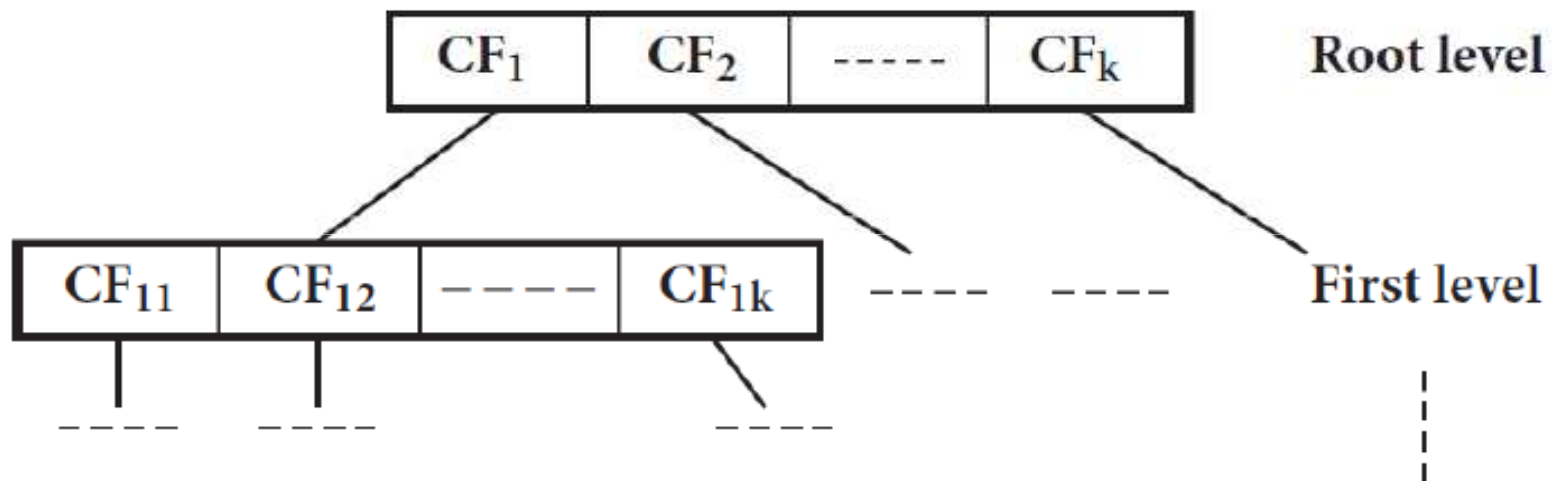
$$\begin{aligned} CF_1 &= \langle 3, (2 + 3 + 4, 5 + 2 + 3), (2^2 + 3^2 + 4^2, 5^2 + 2^2 + 3^2) \rangle \\ &= \langle 3, (9, 10), (29, 38) \rangle. \end{aligned}$$

- Suppose that C_1 is joint 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:

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

BIRCH Algorithm

- A **CF tree** is a height-balanced tree that stores the clustering features for a hierarchical clustering.



BIRCH Algorithm

- By definition, a nonleaf node in a tree has **children**.
- The nonleaf nodes store sums of the CFs of their children, and thus summarize clustering information about their children.
- **A CF tree has two parameters:**
 - **Branching factor, B**
 - ◆ specifies the maximum number of children per nonleaf node.
 - **Threshold, T**
 - ◆ specifies the maximum diameter of subclusters stored at the leaf nodes of the tree.
- These two parameters influence the size of the resulting tree.

BIRCH Algorithm

- BIRCH applies a multiphase clustering technique
 - A single scan of the data set yields good clustering and one or more additional scans can be used to improve the quality
- **Phase 1:** BIRCH scans the db to build an initial in-memory CR-tree which preserves the data's inherent clustering technique.
- **Phase 2:** BIRCH applies a clustering algorithm to cluster the leaf nodes of the CF-tree which removes sparse clusters as outliers and groups dense clusters into large ones.



BIRCH Algorithm Phases

● Phase 1:

- the CF tree is built dynamically as objects are inserted.
- Thus, the method is **incremental**.
- An object is inserted into the closest leaf entry (subcluster).
- If the diameter of the subcluster stored in the leaf node after insertion is larger than the threshold value, then the leaf node and possibly other nodes are split.
- After the insertion of the new object, information about it is passed toward the root of the tree.
- The size of the CF tree can be changed by modifying the threshold.

BIRCH Algorithm Phases

- **Phase 2:**

- Once the CF tree is built, any clustering algorithm, such as a typical partitioning algorithm, can be used with the CF tree in Phase 2.

Computational Complexity of the Algorithm

- The computation complexity of the algorithm is $O(n)$,
 - where n is the number of objects to be clustered.
- Experiments have shown the linear scalability of the algorithm with respect to the number of objects and good quality of clustering of the data.

Weakness of BIRCH

- However, since each node in a CF tree can hold only a limited number of entries due to its size, a CF tree node does not always correspond to what a user may consider a natural cluster.
- Moreover, if the clusters are not spherical in shape, BIRCH does not perform well, because it uses the notion of radius or diameter to control the boundary of a cluster.

References

- J. Han, M. Kamber, **Data Mining: Concepts and Techniques**, Elsevier Inc. (2006). (Chapter 7)