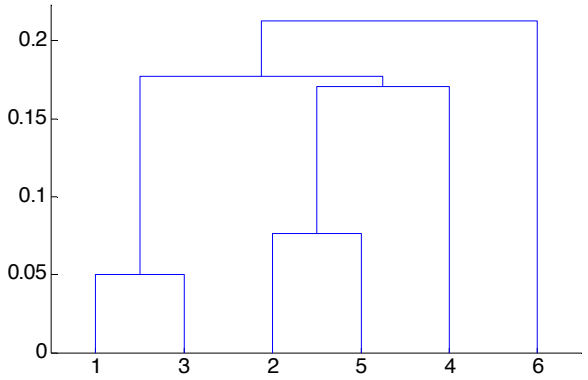
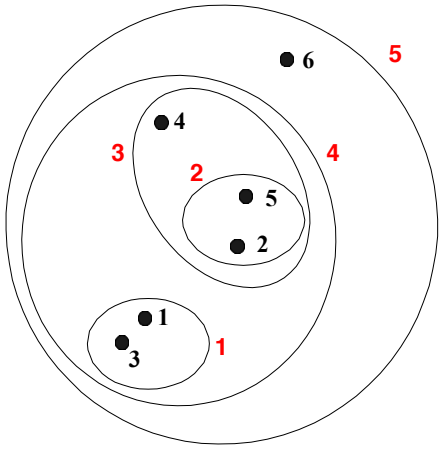
Hierarchical Clustering

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
  - Any desired number of clusters can be obtained by 'cutting' the dendogram at the proper level
- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

An Example

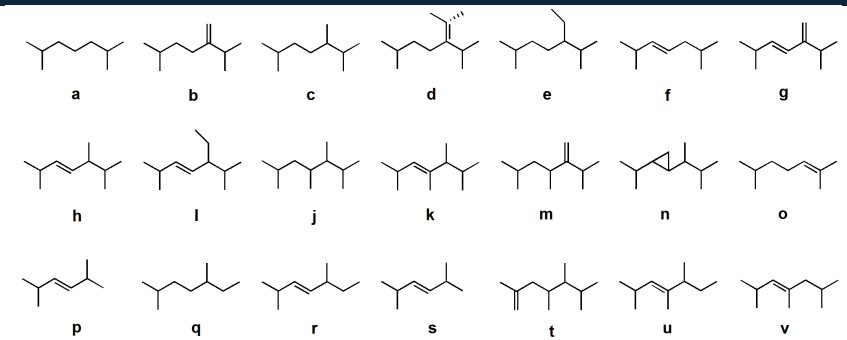
- Study: How different species of Dinoflagellates (Algae) relates to each other by studying their sterol composition
  - identify the relationships via sterol composition similarity amongst dinoflagellates
  - investigate the correspondences between the dinoflagellates sharing a similar sterol compositions and their evolutionary histories.

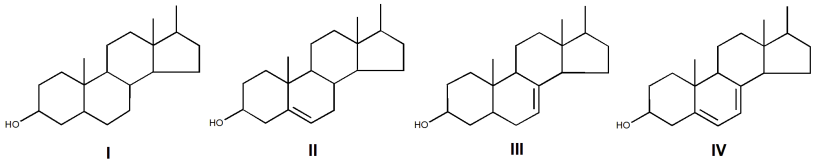
An Example

- Data:
  - 58 named sterols and steroidal ketones
  - 102 dinoflagellate species
- Analysis method
  - Hierarchical Clustering based on Sterol composition data
  - Clustering validation using multiple clustering schemes and clustering validation criteria

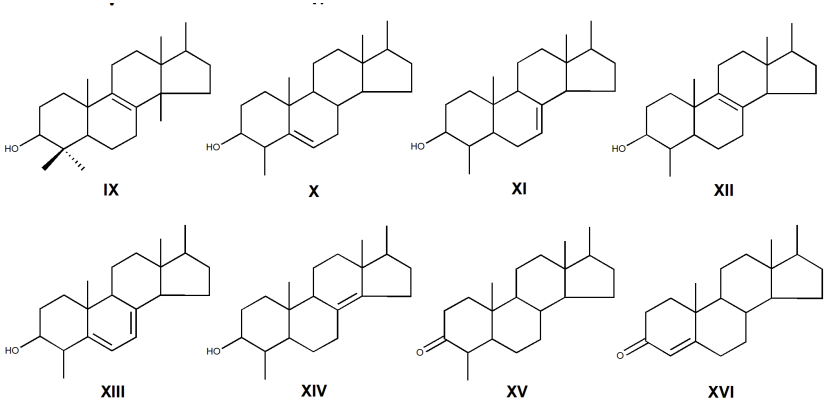
An Example

Structures of sterols in the study

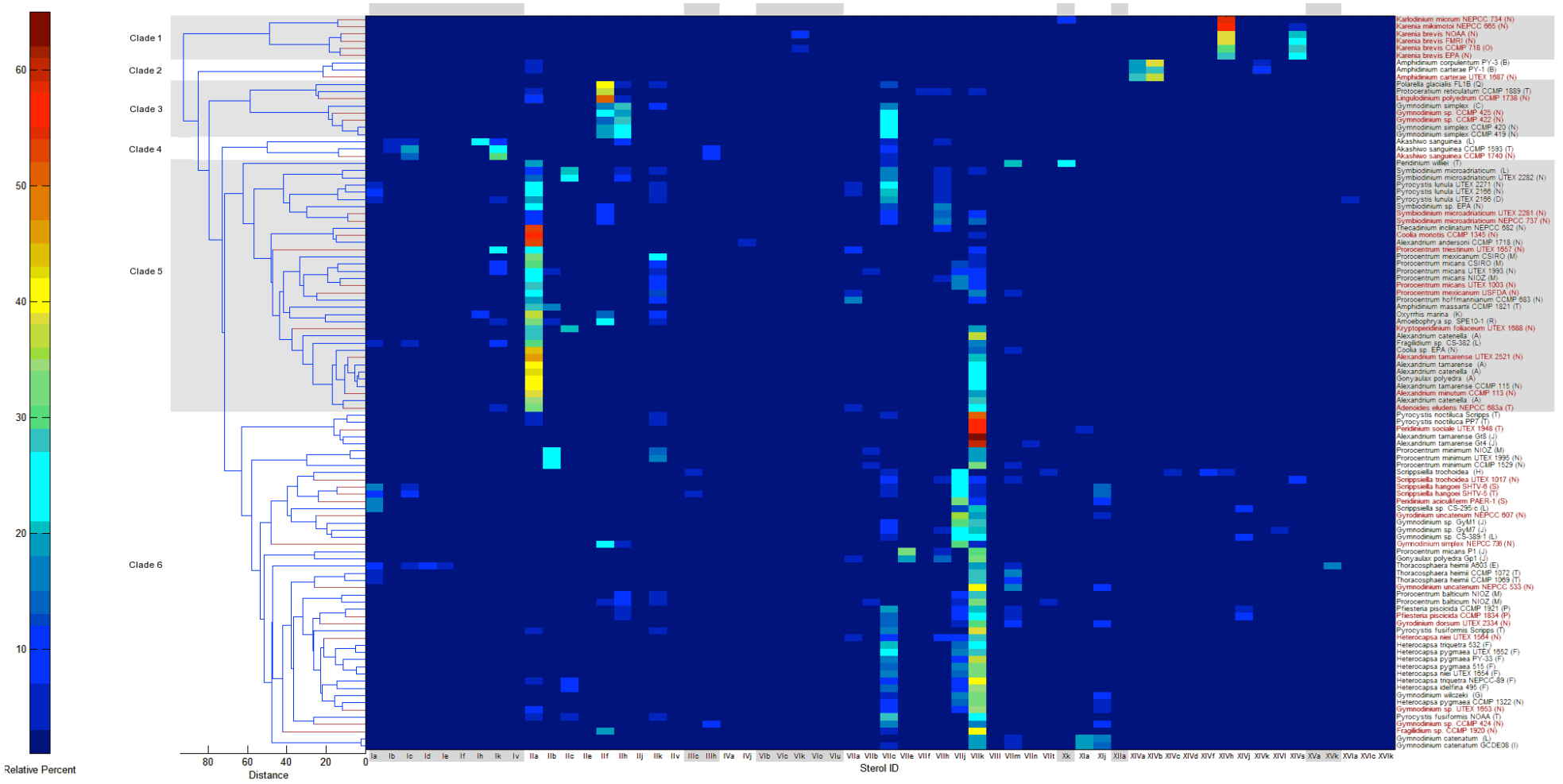




	5α-Cholestan-3β-ol (cholestanol)	I a
	24-Methylcholesta-24(28)-en-3β-ol (24-methylenecholestanol)	I b
	24-Methyl-5α-cholestan-3β-ol	I c
HO	24-Ethyl-5α-cholest-24(28)E-en-3β-ol	I d

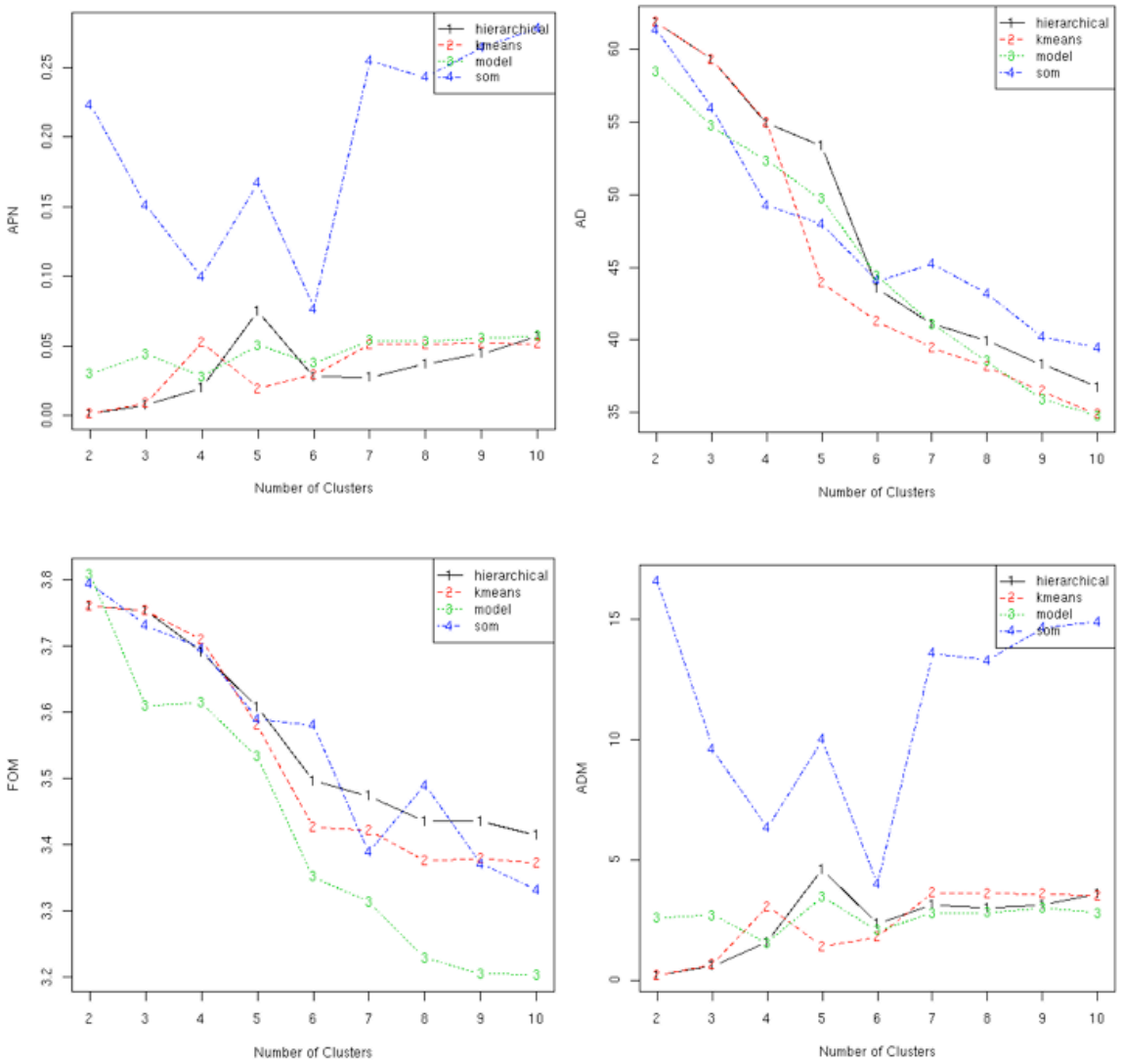


Hierarchical Clustering Result



Dendrogram of dinoflagellate relationships based on sterol compositions accompanied by heat map showing sterol distributions

Clustering Validation



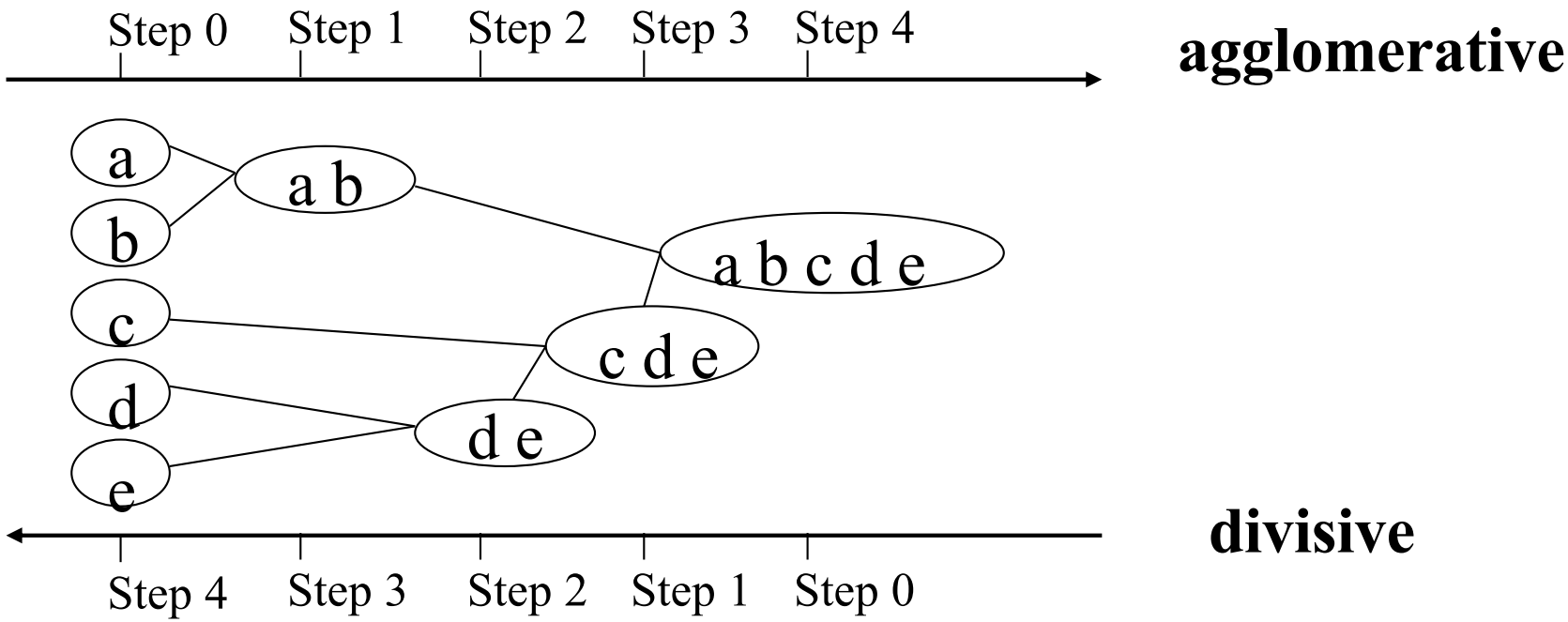
Cluster validation on sterol data using the Internal and the stability measures

An Example

- Conclusions of the Study:
  - Our results indicated that several, but not all, dinoflagellate genera share similar sterol compositions
  - Sterol composition of dinoflagellates has been determined, to a certain extent, by the evolutionary diversification of this lineage.

Agglomerative vs. Divisive 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



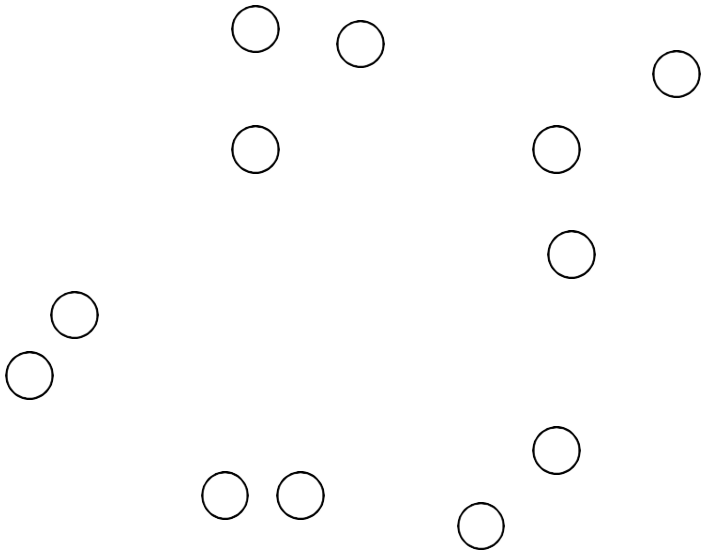
Agglomerative Clustering

- More popular hierarchical clusteringtechnique
- Basic algorithm isstraightforward
  1. Compute the proximitymatrix
  2. Let each data point be a cluster
  3. Repeat
  4. Merge the two closest clusters
  5. Update the proximitymatrix
  6. Until only a single cluster remains
- Key operation is the computation of the proximity of two clusters

Different approachesto defining the distance between clusters distinguish the different algorithms

Starting Situation

- Start with clusters of individual points and a proximity matrix

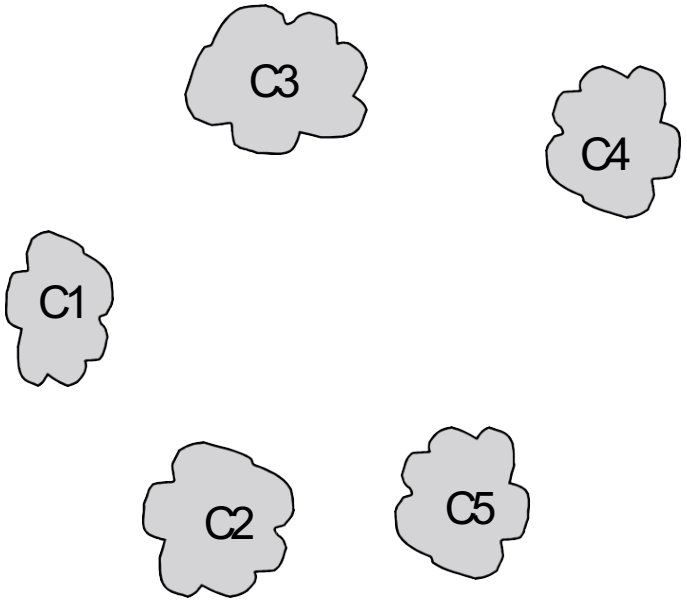


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

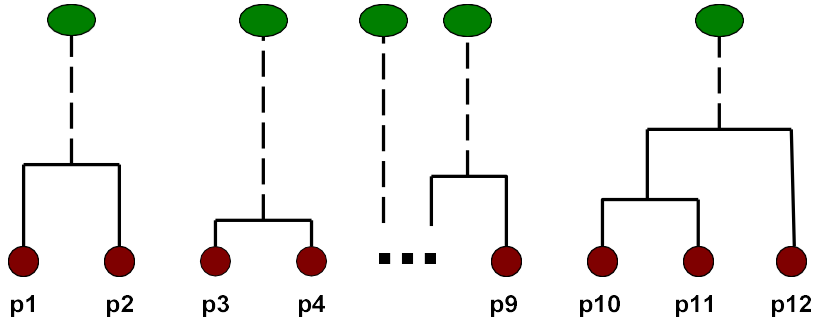
Intermediate Situation

- After some merging steps, we have some clusters



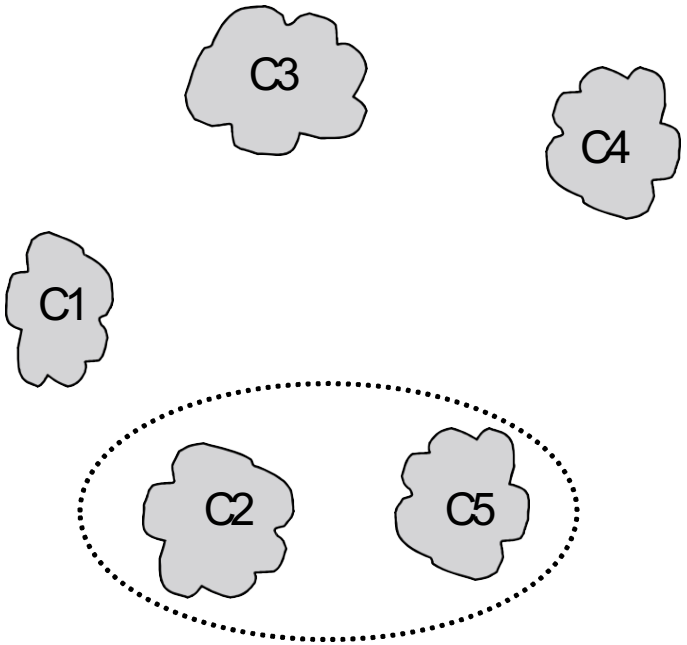
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

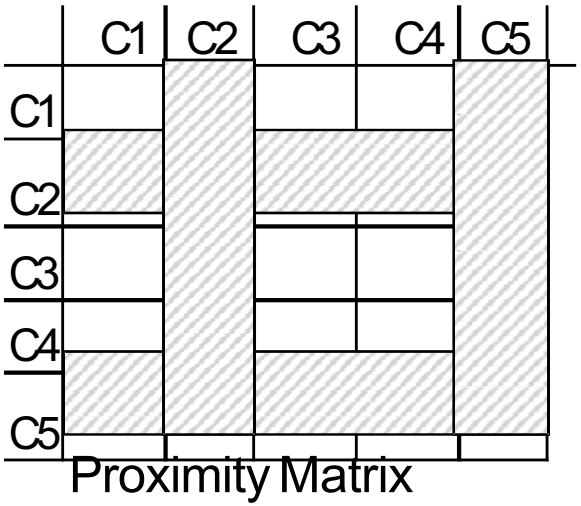
Proximity Matrix

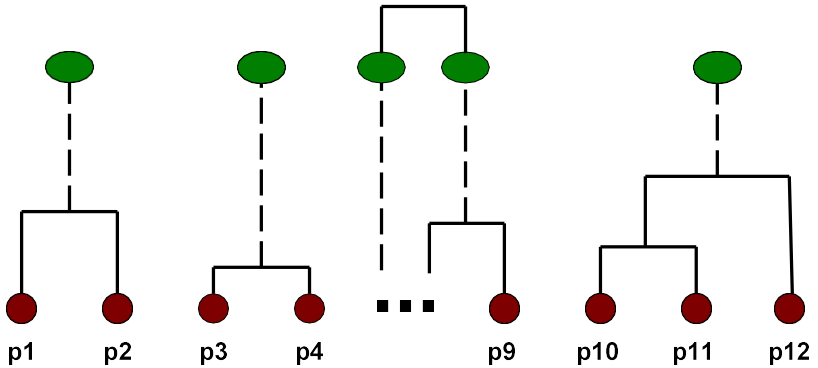


Intermediate Situation

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

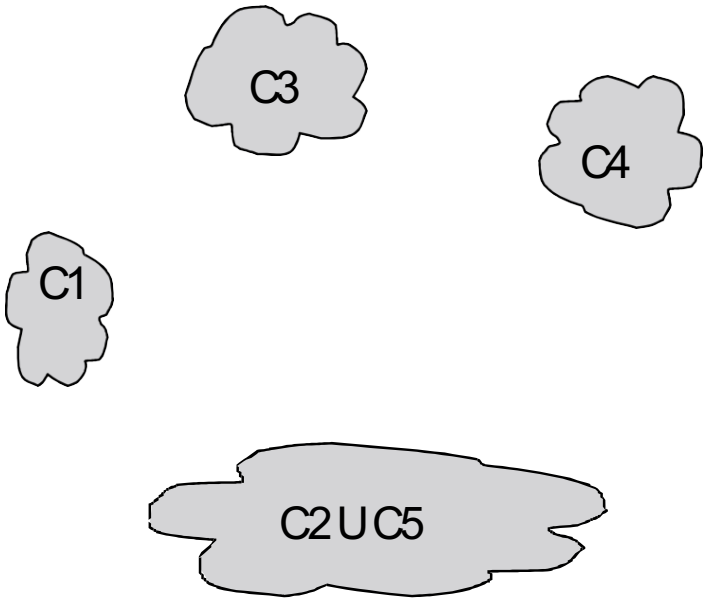






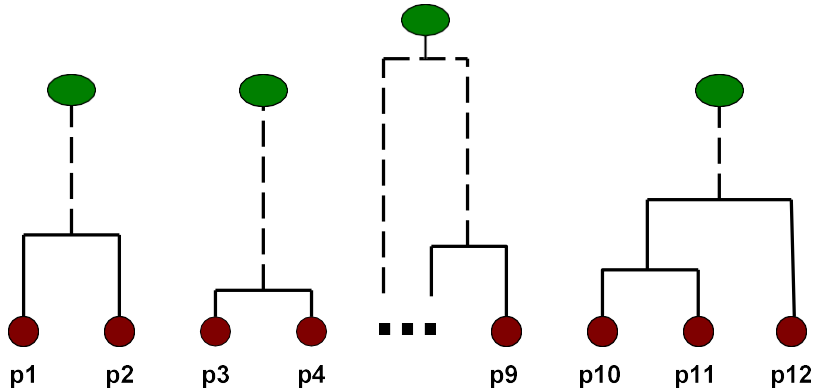
After the Merging

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

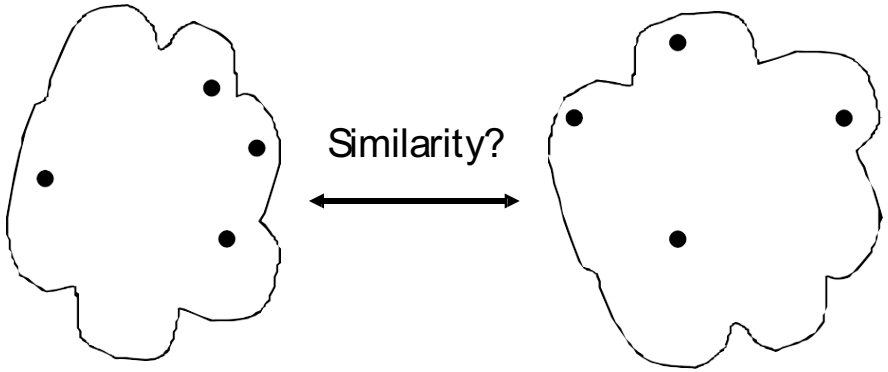


Proximity Matrix

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



Define Inter-cluster Similarity

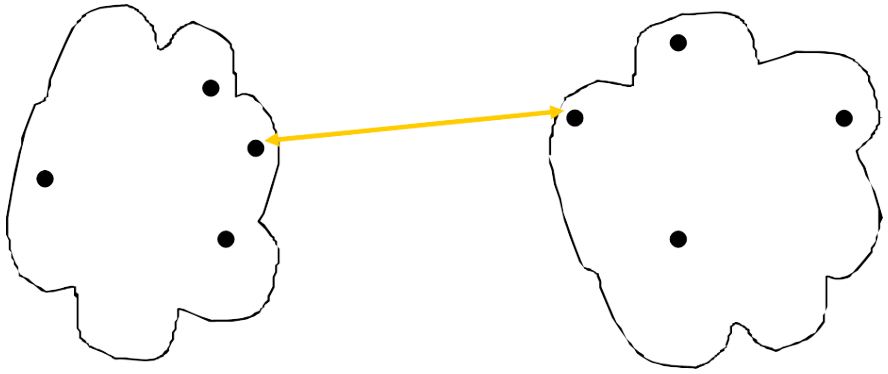


- MIN
- MAX
- Group Average
- Distance Between on Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

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

Proximity Matrix

Define Inter-cluster Similarity

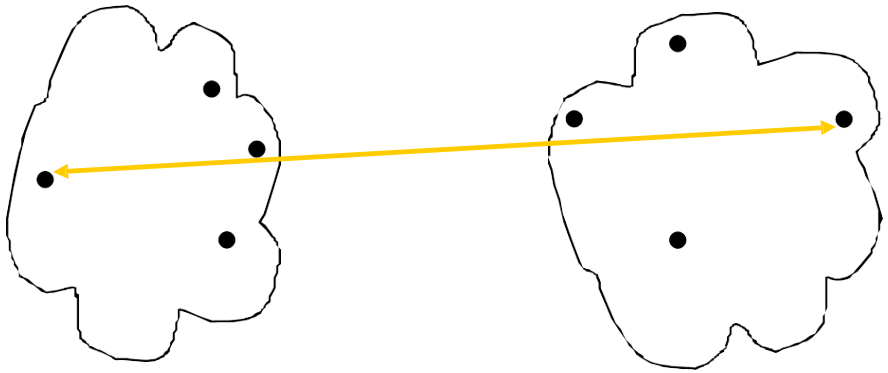


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

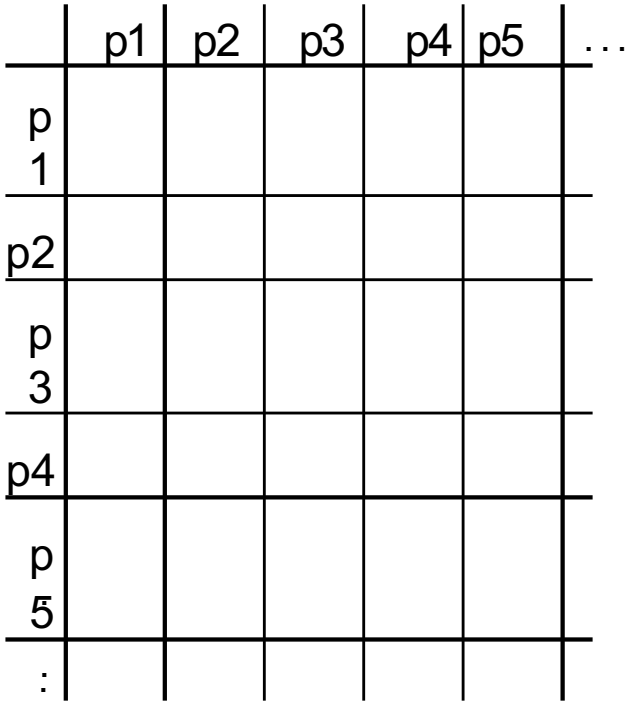
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
:						

Proximity Matrix

Define Inter-cluster Similarity

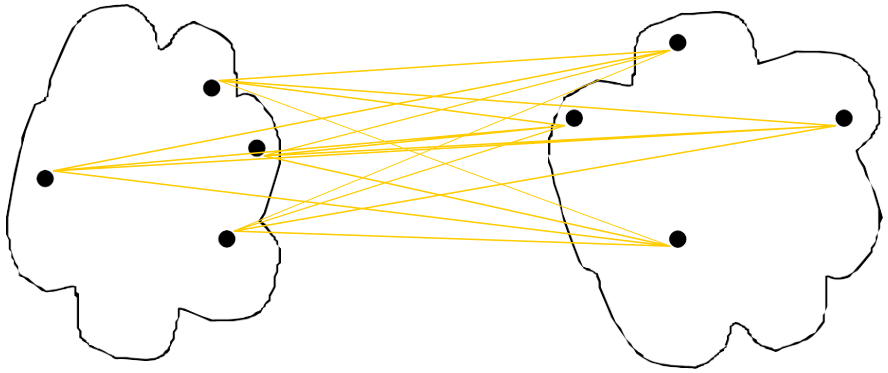


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error



Proximity Matrix

Define Inter-cluster Similarity

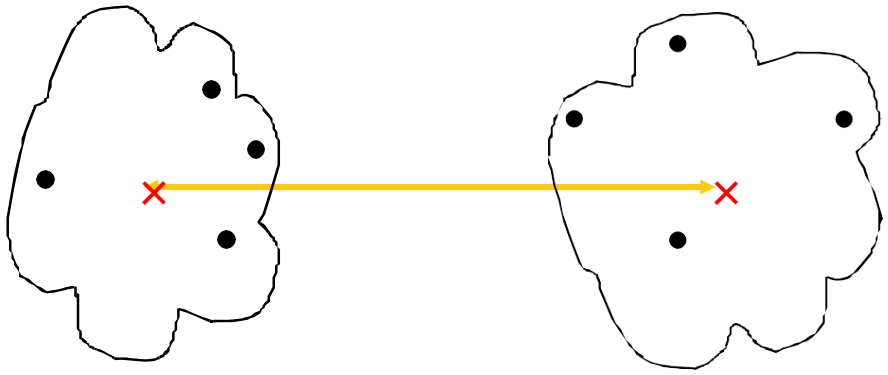


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

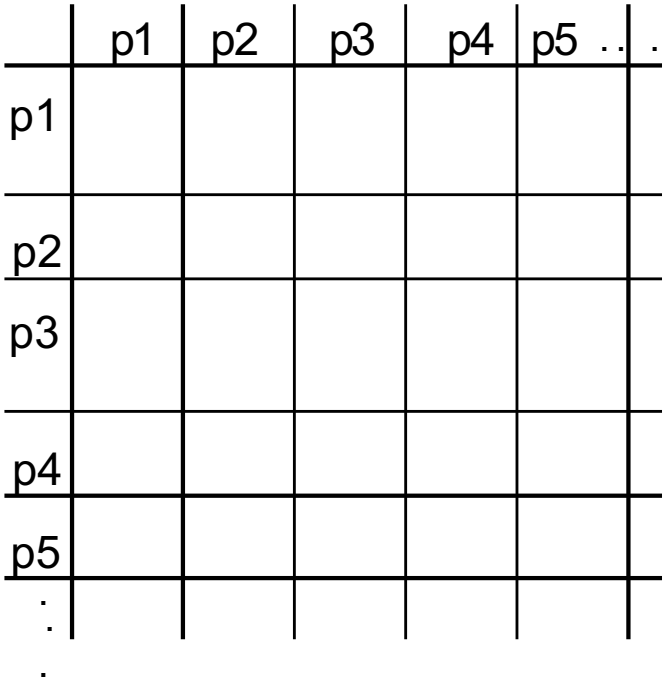
Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

Define Inter-cluster Similarity



- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

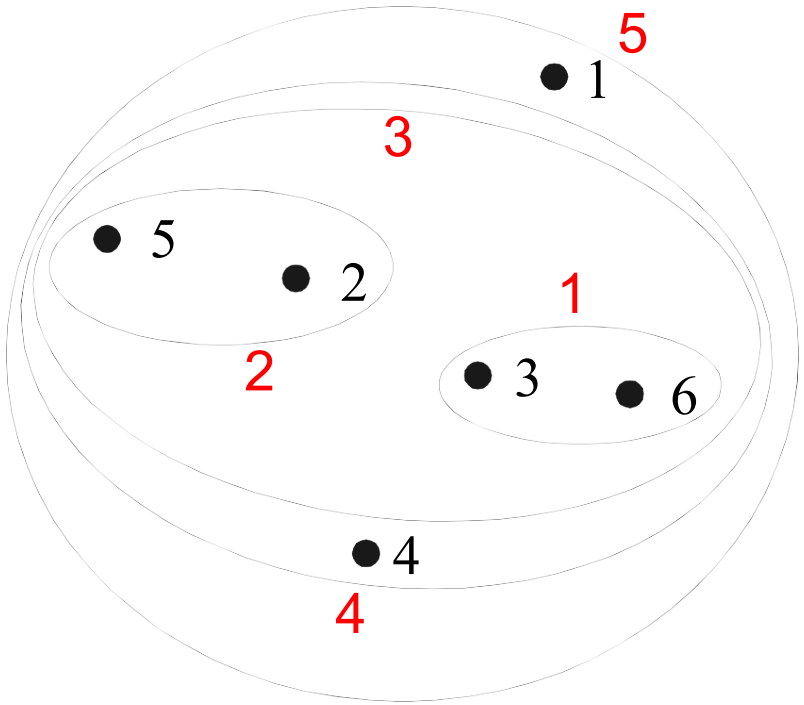


Proximity Matrix

Single link vs. Complete link

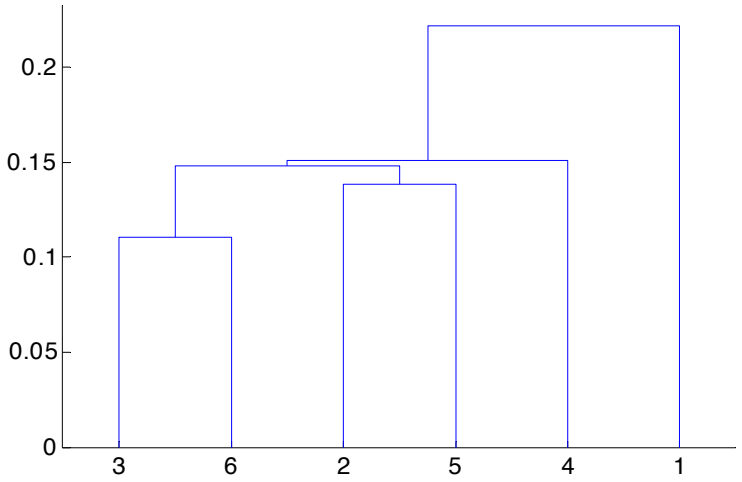
- Another way to view the processing of the hierarchical algorithm is that we create links between their elements in order of increasing distance
  - The MIN – Single Link, will merge two clusters when a single pair of elements is linked
  - The MAX – Complete Linkage will merge two clusters when all pairs of elements have been linked.

Single Link - Min



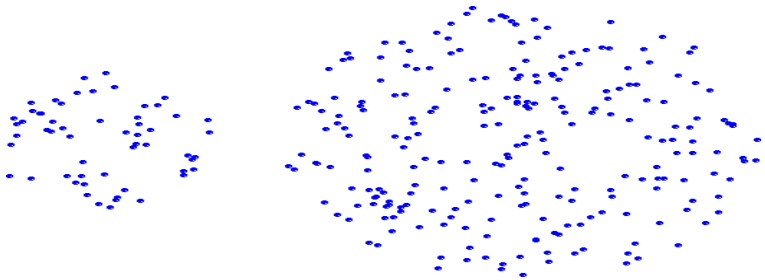
Nested Clusters

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0

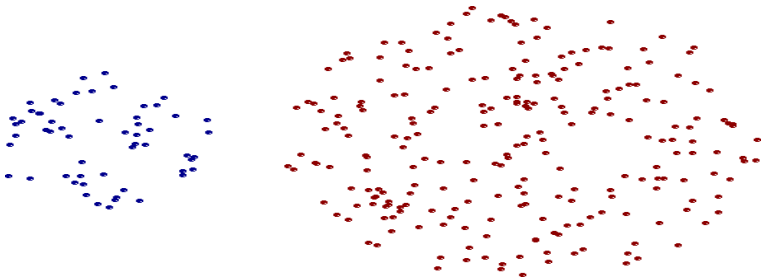


Dendrogram

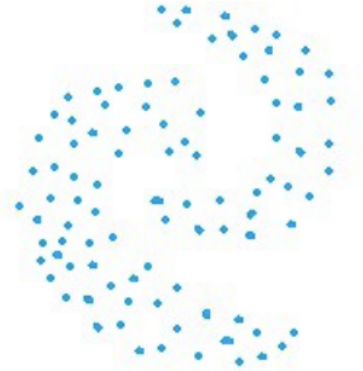
Strength of Single Link

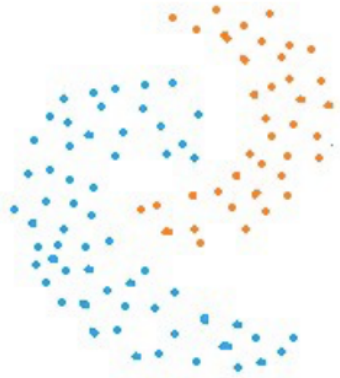


Original Points



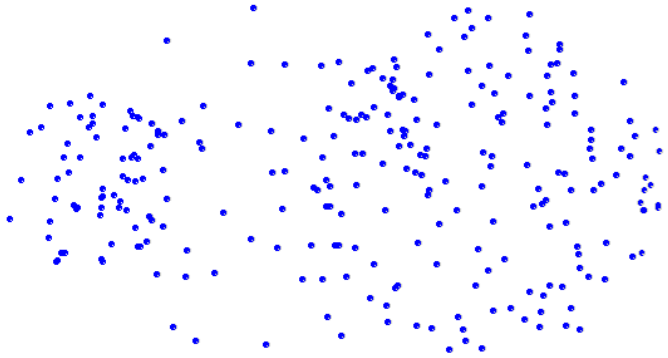
Two Clusters



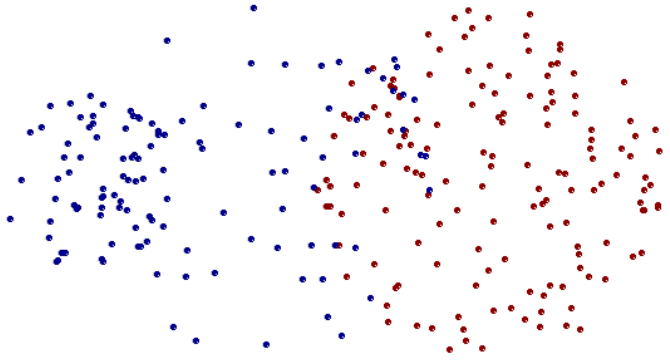


Can handle non-elliptical shapes

Limitation of Single Link



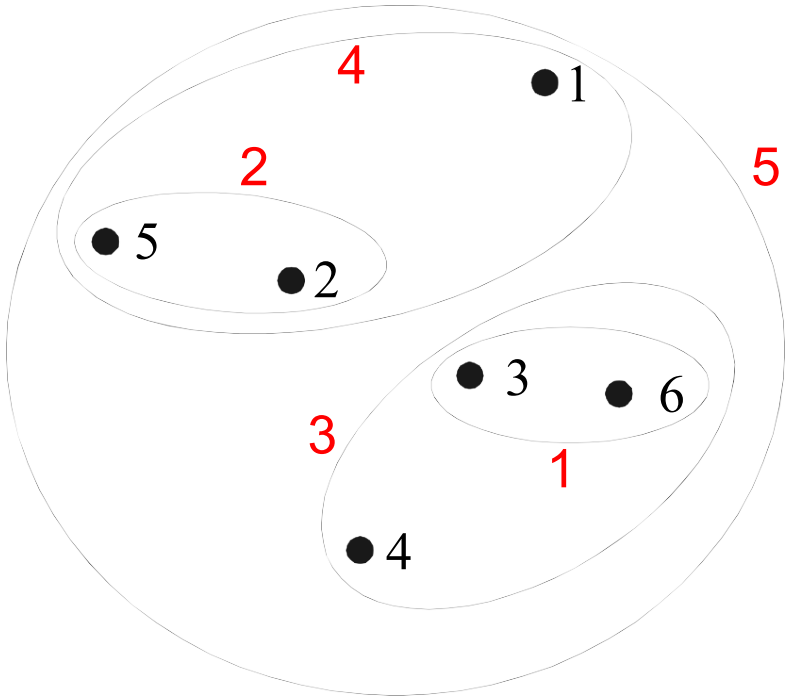
Original Points



Two Clusters

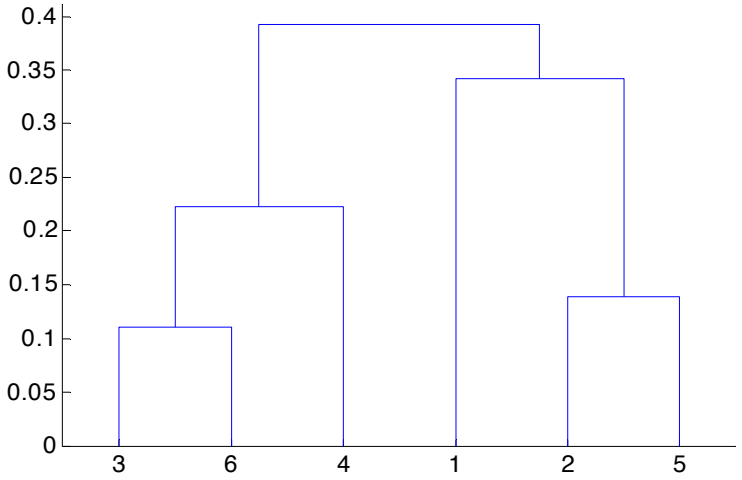
Sensitive to noise and outliers

Complete Link - Max



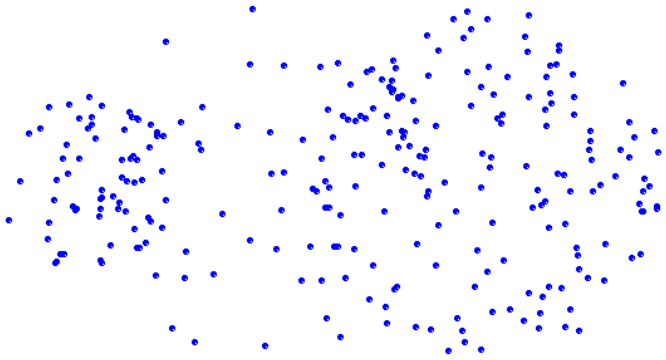
Nested Clusters

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0

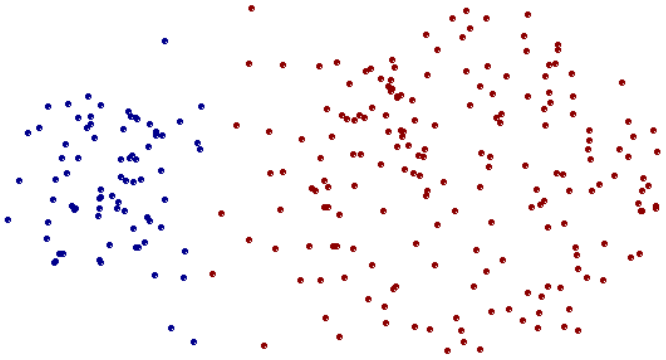


Dendrogram

Strength of Complete Link



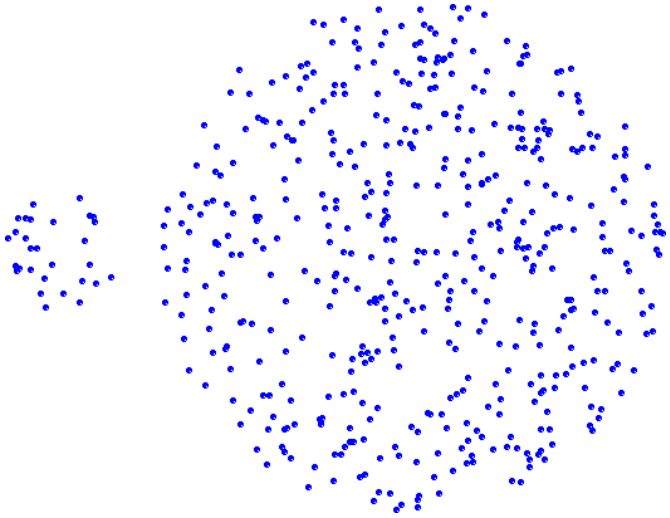
Original Points



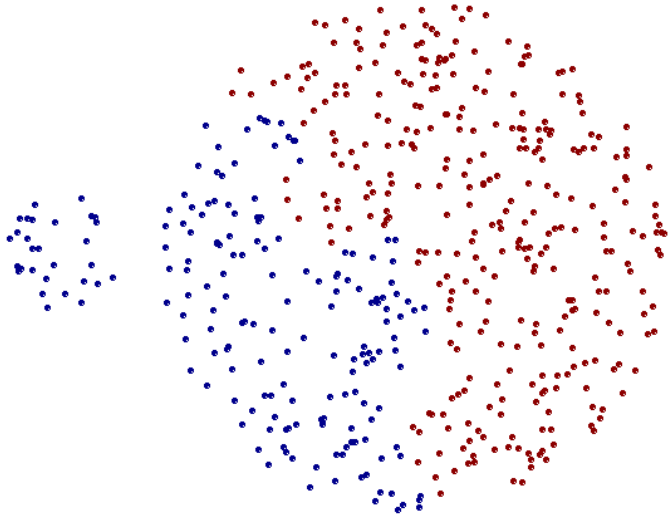
Two Clusters

Less susceptible to noise and outliers

Limitation of Complete Link



Original Points



Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

Hierarchical Clustering: Group Average

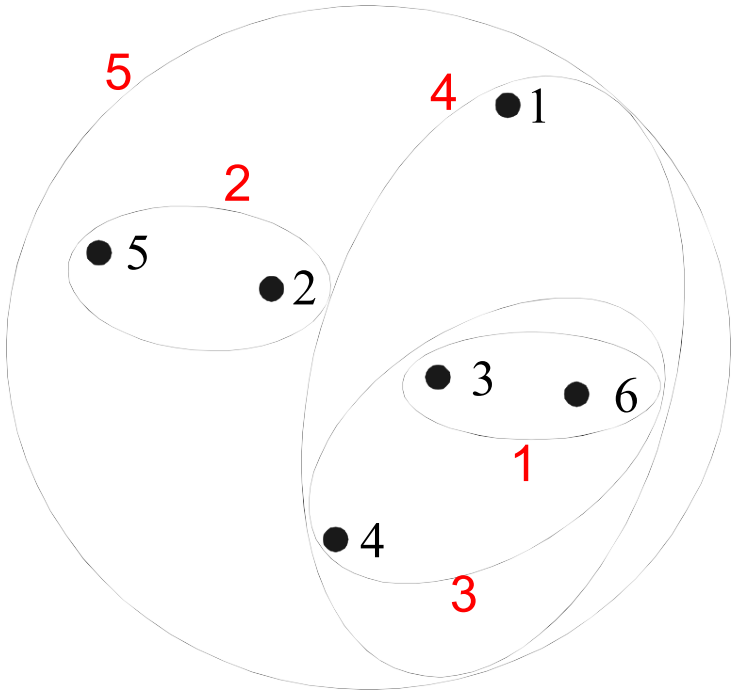
- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

- Need to use average connectivity for scalability since total proximity favors large clusters

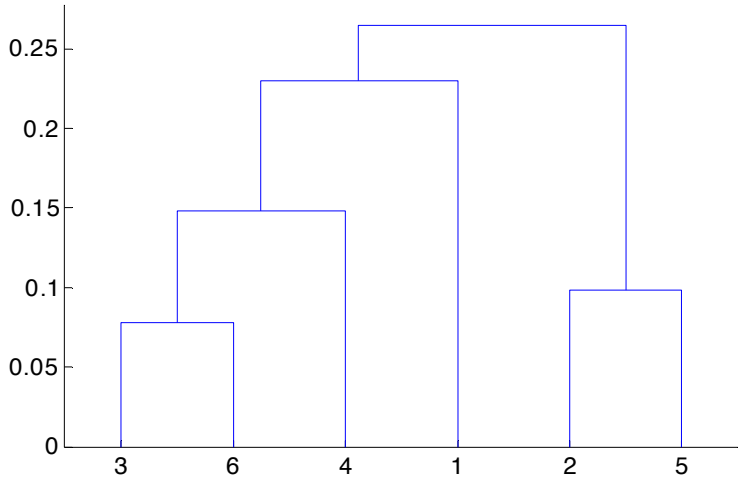
	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0

Hierarchical Clustering: Group Average



Nested Clusters

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0



Dendrogram

Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise and outliers
- Limitations
  - Biased towards globular clusters

Ward's Method

- Similarity of two clusters is based on the increase in squared error (SSE) when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used to initialize K-means

Time and Space Complexity

- $O(N^2)$ 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 for some approaches

Problems and Limitations

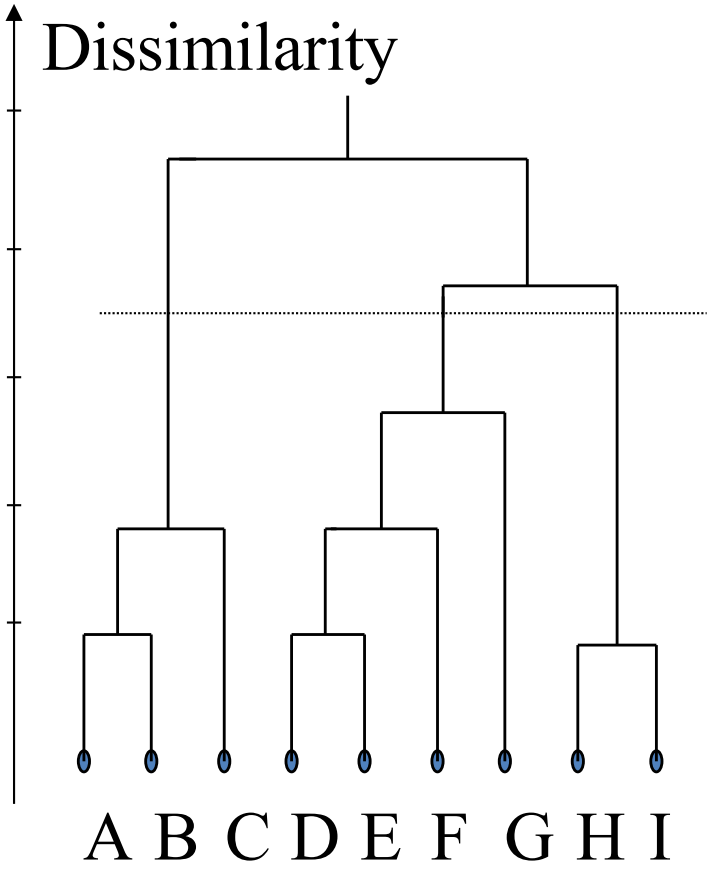
- Computational complexity in time and space
- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers
  - Difficulty handling different sized clusters and convex shapes
  - Breaking large clusters

Dendrogram

A *Dendrogram* Shows How the Clusters are Merged Hierarchically:

- Decompose data objects into several levels of nested partitioning (tree of clusters), called a dendrogram.

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



Practice Question

- Apply the single-link, complete link, average link, or Ward's agglomerative clustering methods to cluster the following six objects :

	Gender	Age	Time	Fever	Cough
Obj1	F	2	2	Y	N
Obj2	M	2	0.5	N	N
Obj3	F	15	3	Y	Y
Obj4	F	18	0.5	Y	N
Obj5	M	58	4	N	Y
Obj6	F	44	14	N	Y

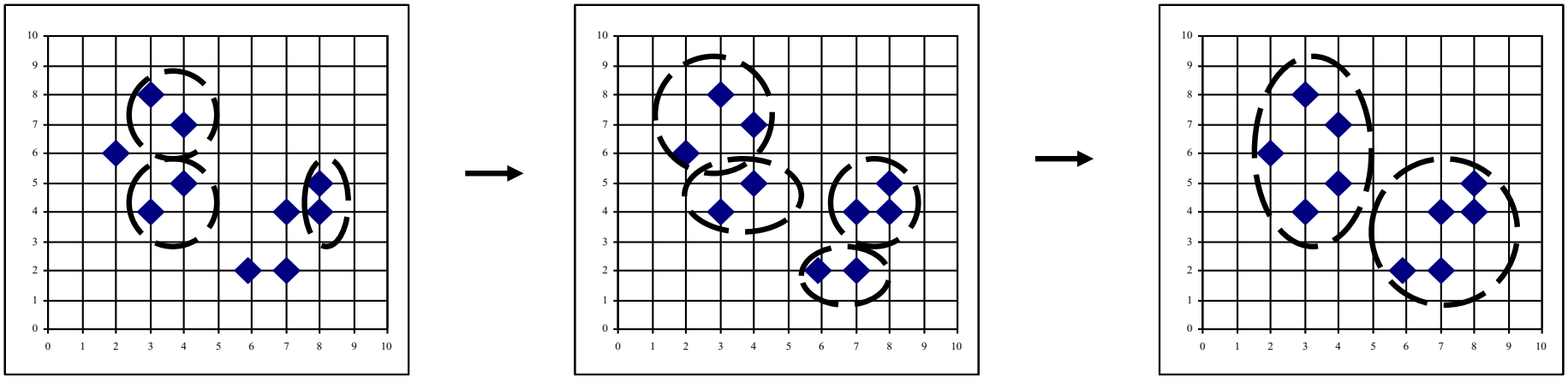
Practice Question

The distance(dissimilarity) between pairwise objects

	O1	O2	O3	O4	O5	O6
O1	--					
O2	0.94	--				
O3	0.36	0.91	--			
O4	0.19	0.75	0.39	--		
O5	1.15	1.38	0.99	1.3	--	
O6	1.38	2.16	1.22	1.5	1.2	--

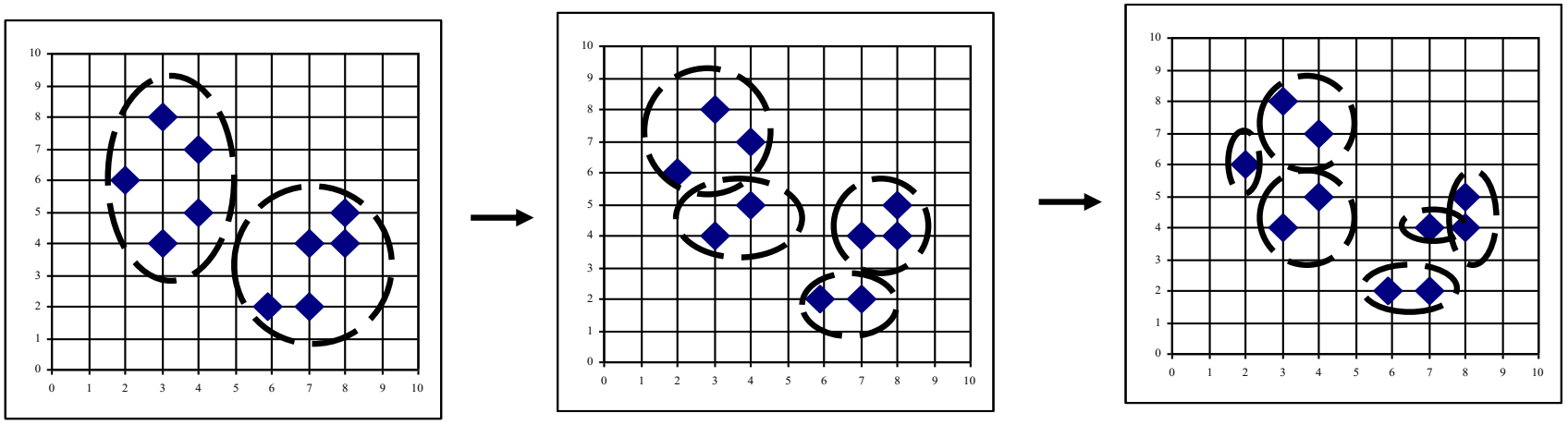
AGNES (Agglomerative Nesting)

- Implemented in statistical analysis packages, e.g., Splus
- Use the Single-Link method and the dissimilarity matrix.
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster



DIANA (Divisive Analysis)

- Inverse order of AGNES
- Eventually each node forms a cluster on its own



More on Hierarchical Clustering Methods

- **Major weakness of agglomerative clustering methods**
  - do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
  - can never undo what was done previously
- **Integration of hierarchical with distance-based clustering**
  - BIRCH (1996) (Balanced Iterative Reducing and Clustering using Hierarchies): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - CURE (1998): selects well-scattered points from the cluster and then shrinks them towards the center of the cluster by a specified fraction
  - …

Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)

- 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.

Clustering Feature Vector

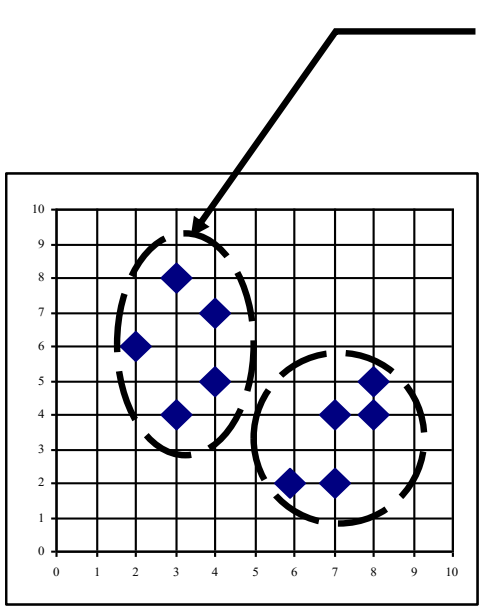
Used to compute centroids, and measures the compactness and distance of clusters

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

N: **Number of data points**

LS: $\sum_{i=1}^{N} \vec{X_i}$

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



Centroid, radius, diameter

Given a cluster of instances $\{\vec{X}_i\}$*, we define the* ***centroid****, the* ***radius****, and the* ***diameter****:*

The centroid is the center of the cluster

$$\vec{X0} = \frac{\sum_{i=1}^{N} \vec{X}_i}{N}$$

The radius is average distance of each instance to the centroid

$$R = \left(\frac{\sum_{i=1}^{N} (\vec{X}_i - \vec{X0})^2}{N}\right)^{\frac{1}{2}}$$

The diameter is the average distance between any two data within a cluster

$$D = \left(\frac{\sum_{i=1}^{N} \sum_{j=1}^{N} (\vec{X}_i - \vec{X}_j)^2}{N(N-1)}\right)^{\frac{1}{2}}$$

← Key criterion used to control the height of the tree

Some distance measures

Euclidean *and* ***Manhattan*** *distance between any two clusters:*

$$D0 = ((\vec{X}_{O_1} - \vec{X}_{O_2})^2)^{\frac{1}{2}}$$

$$D1 = \left|\vec{X}_{O_1} - \vec{X}_{O_2}\right| = \sum_{i=1}^{d} \left|\vec{X}_{O_1}^{(i)} - \vec{X}_{O_2}^{(i)}\right|$$

Some distance measures

*Define the **average inter-cluster**, the **average intra-cluster**, and the **variance increase** distances as:*

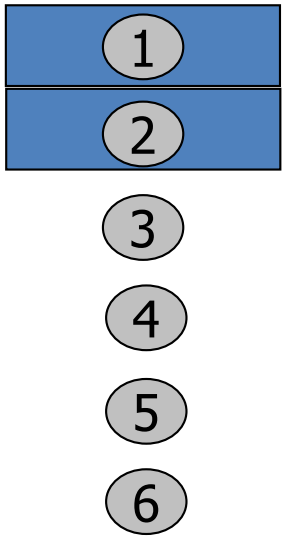
$$D2 = \left(\frac{\sum_{i=1}^{N_1}\sum_{j=N_1+1}^{N_1+N_2}(\vec{X}_i - \vec{X}_j)^2}{N_1 N_2}\right)^{\frac{1}{2}}$$

$$D3 = \left(\frac{\sum_{i=1}^{N_1+N_2}\sum_{j=1}^{N_1+N_2}(\vec{X}_i - \vec{X}_j)^2}{(N_1+N_2)(N_1+N_2-1)}\right)^{\frac{1}{2}}$$

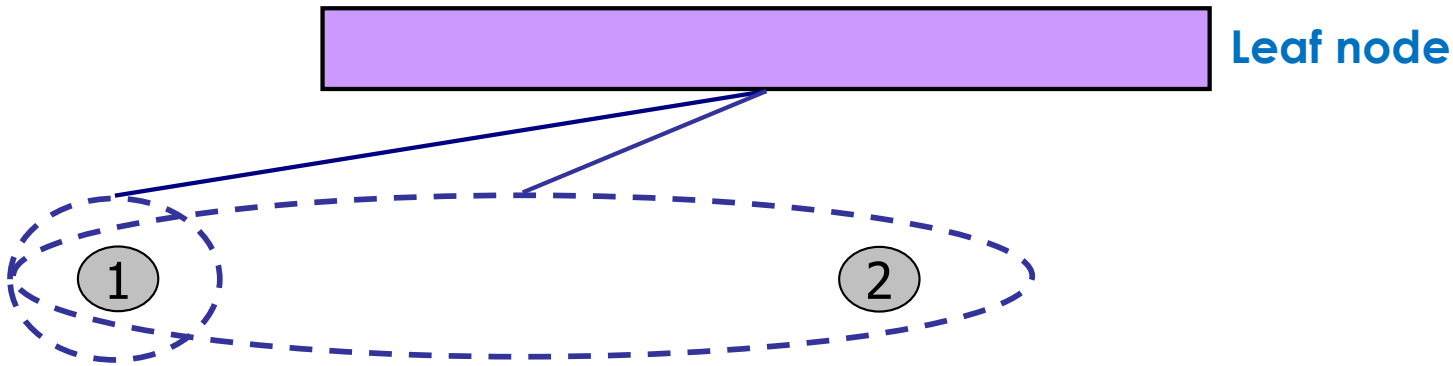
$$D4 = \left(\sum_{k=1}^{N_1+N_2}\left(\vec{X}_k - \frac{\sum_{l=1}^{N_1+N_2}\vec{X}_l}{N_1+N_2}\right)^2 - \sum_{i=1}^{N_1}\left(\vec{X}_i - \frac{\sum_{l=1}^{N_1}\vec{X}_l}{N_1}\right)^2 - \sum_{j=N_1+1}^{N_1+N_2}\left(\vec{X}_j - \frac{\sum_{l=N_1+1}^{N_1+N_2}\vec{X}_l}{N_2}\right)^2\right)^{\frac{1}{2}}$$

BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)

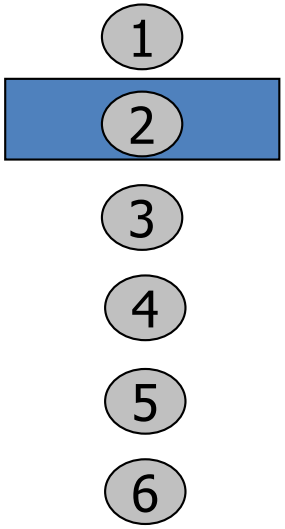


Cluster1

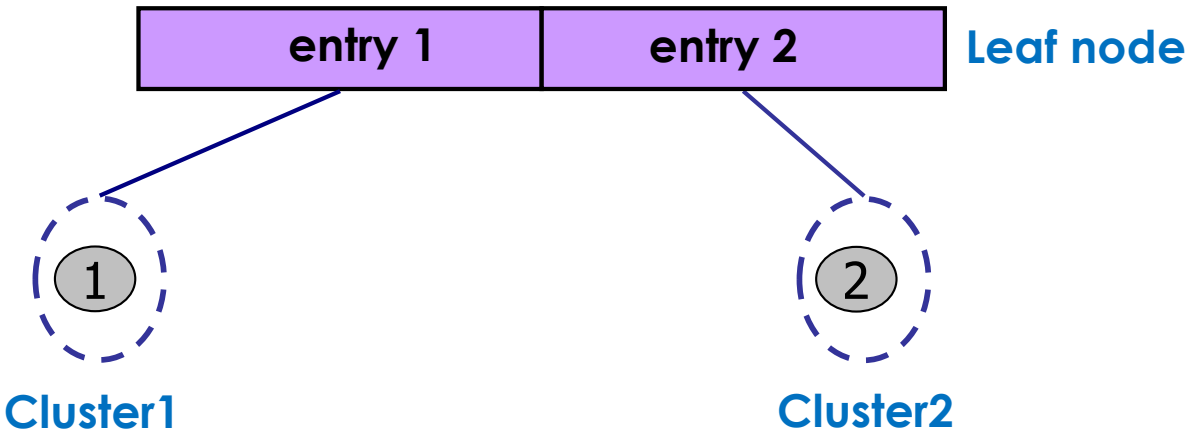
If cluster 1 becomes too large (not compact) by adding object 2, then split the cluster

BIRCH: The Idea by example

Data Objects



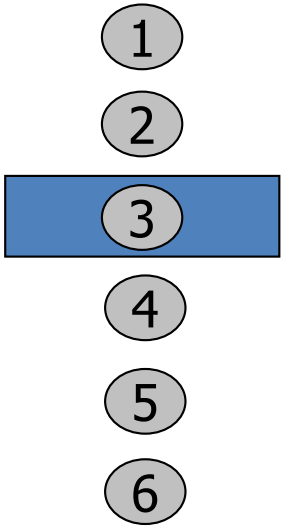
Clustering Process (build a tree)



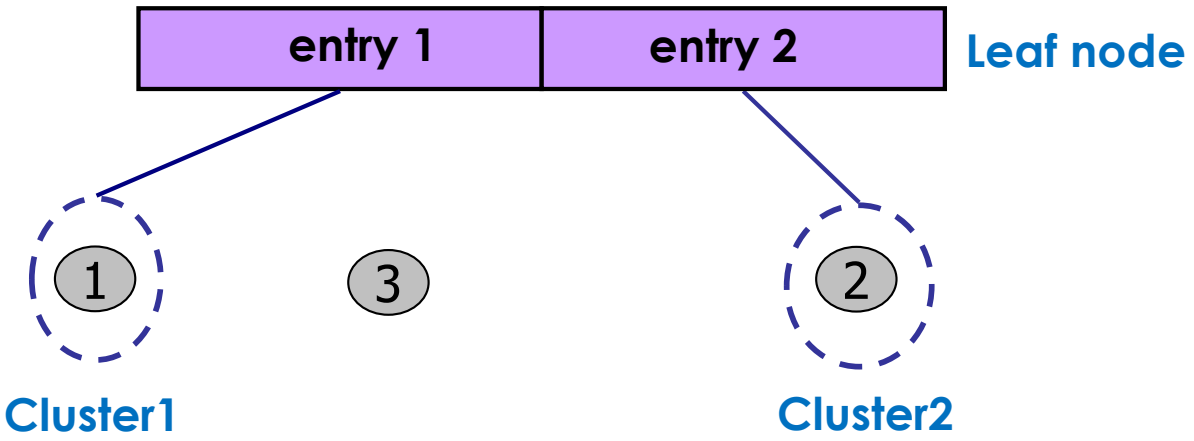
Leaf node with two entries

BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)

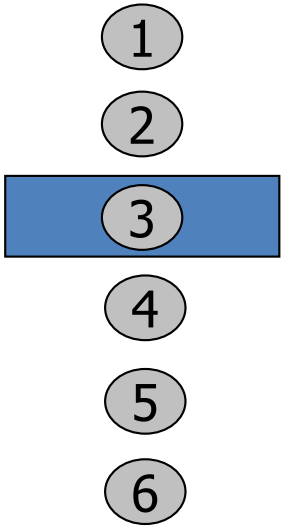


entry1 is the closest to object 3

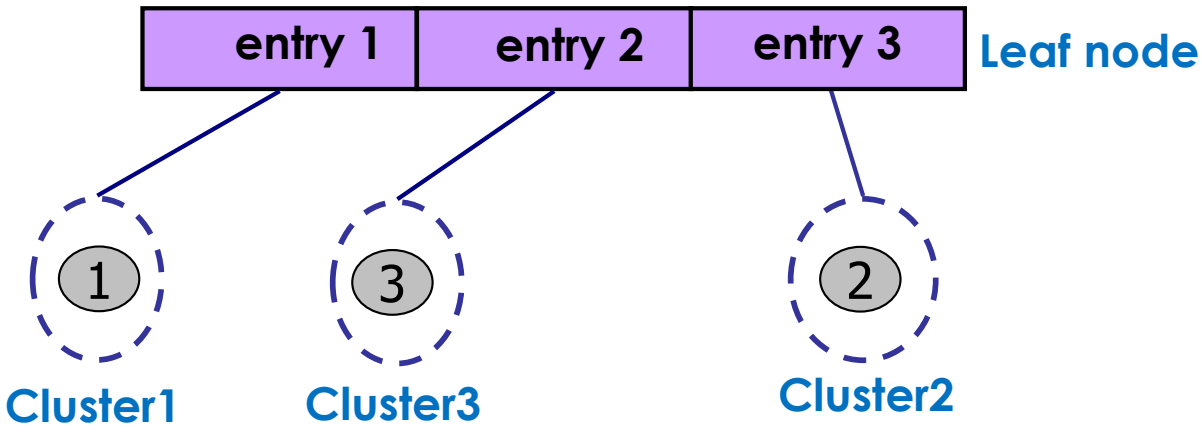
If cluster 1 becomes too large by adding object 3, then split the cluster

BIRCH: The Idea by example

Data Objects



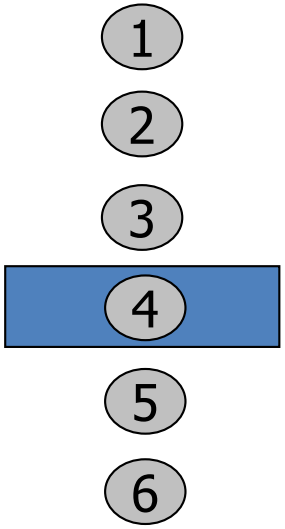
Clustering Process (build a tree)



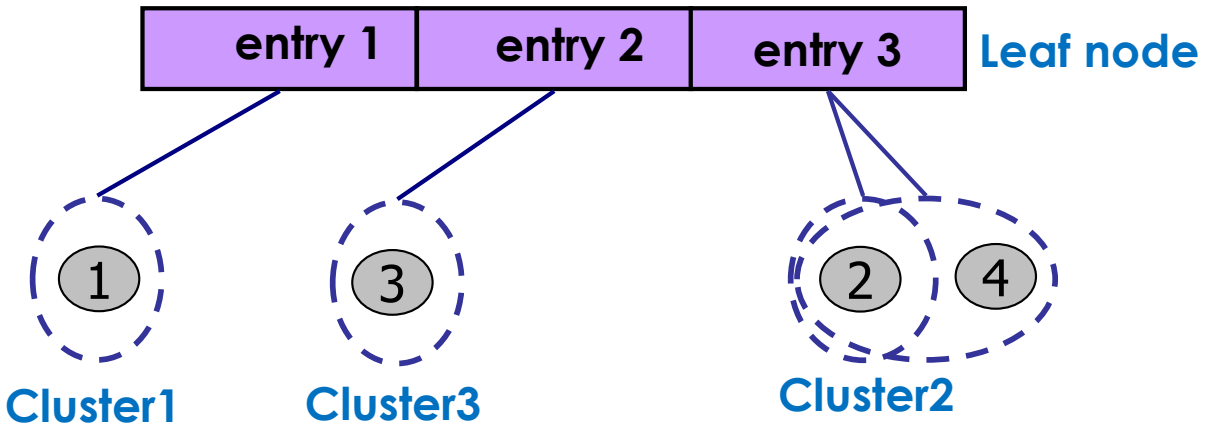
Leaf node with three entries

BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)

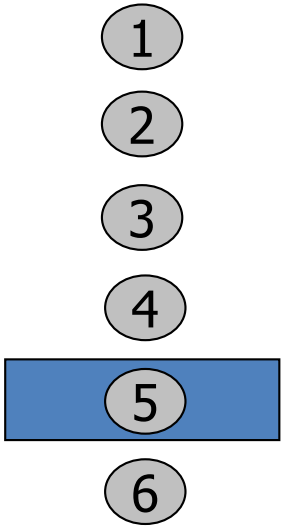


entry3 is the closest to object 4

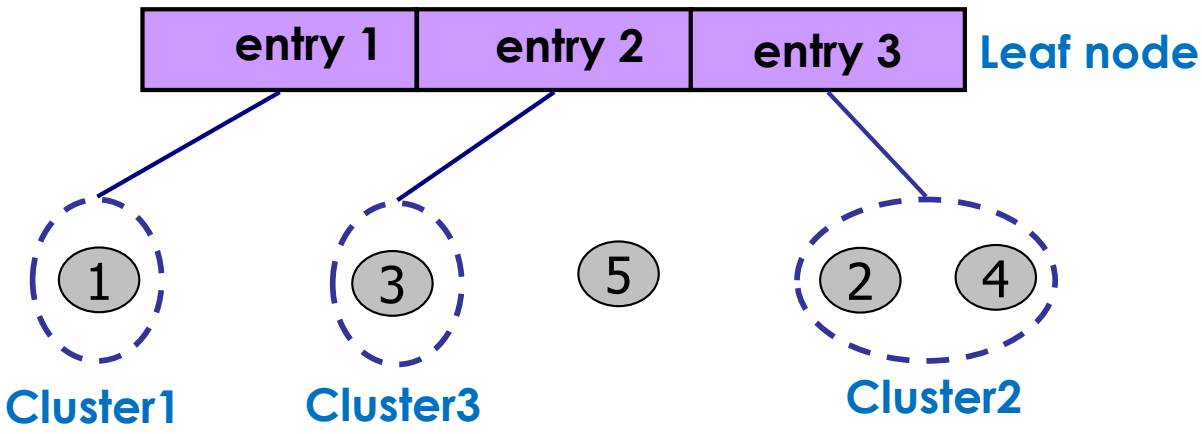
Cluster 2 remains compact when adding object 4 then add object 4 to cluster 2

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?

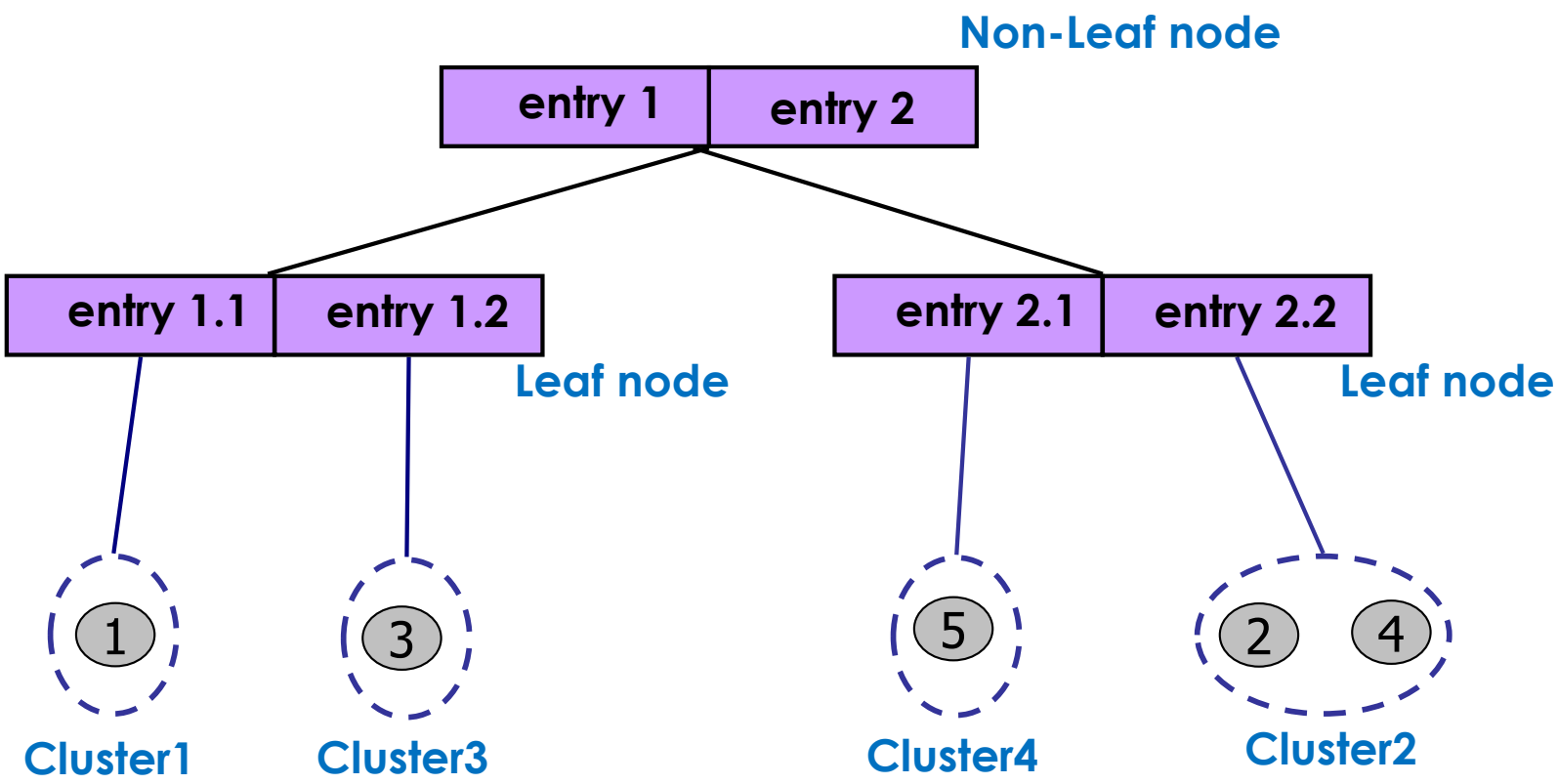
BUT there is a limit to the number of entries a node can have
Thus, split the node

BIRCH: The Idea by example

Data Objects

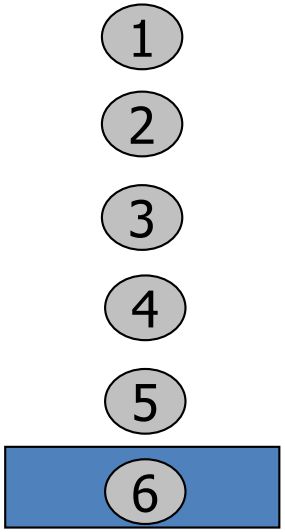
1
2
3
4
5
6

Clustering Process (build a tree)

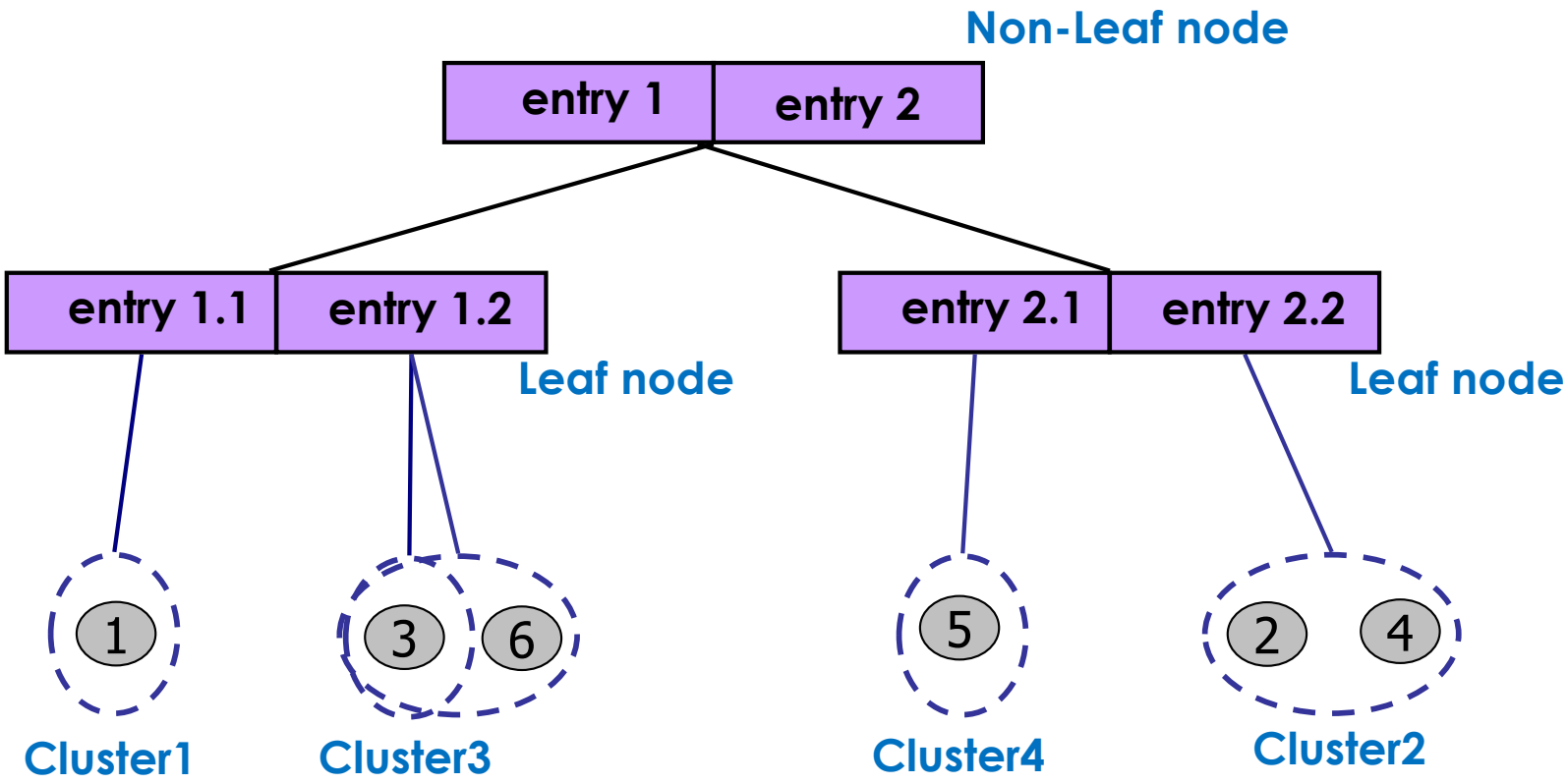


BIRCH: The Idea by example

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

- **CF-Tree**
  - height-balanced tree
  - two parameters:
    - number of entries in each node
    - The *diameter* of all entries in a leaf node
  - Leaf nodes are connected via *prev* and *next* pointers

CF Addictivity

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

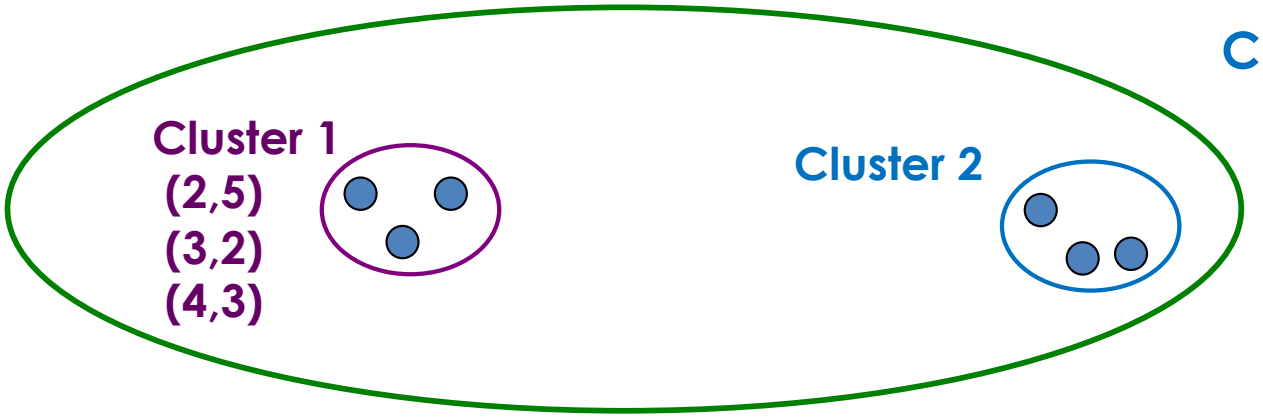
N: Number of data points

LS: linear sum of N points: $\sum_{i=1}^{N} X_i$

SS: square sum of N points: $\sum_{i=1}^{N} X_i^2$

CF3=CF1+CF2= ⟨3+3, (9+35, 10+36), (29+417 , 38+440)⟩ = ⟨6, (44,46), (446 ,478)⟩

Cluster3



CF_2= ⟨3, (35,36), (417 ,440)⟩

CF_1= ⟨3, (2+3+4 , 5+2+3), (2^2+3^2+4^2 , 5^2+2^2+3^2)⟩
= ⟨3, (9,10), (29 ,38)⟩

BIRCH: The Idea by example

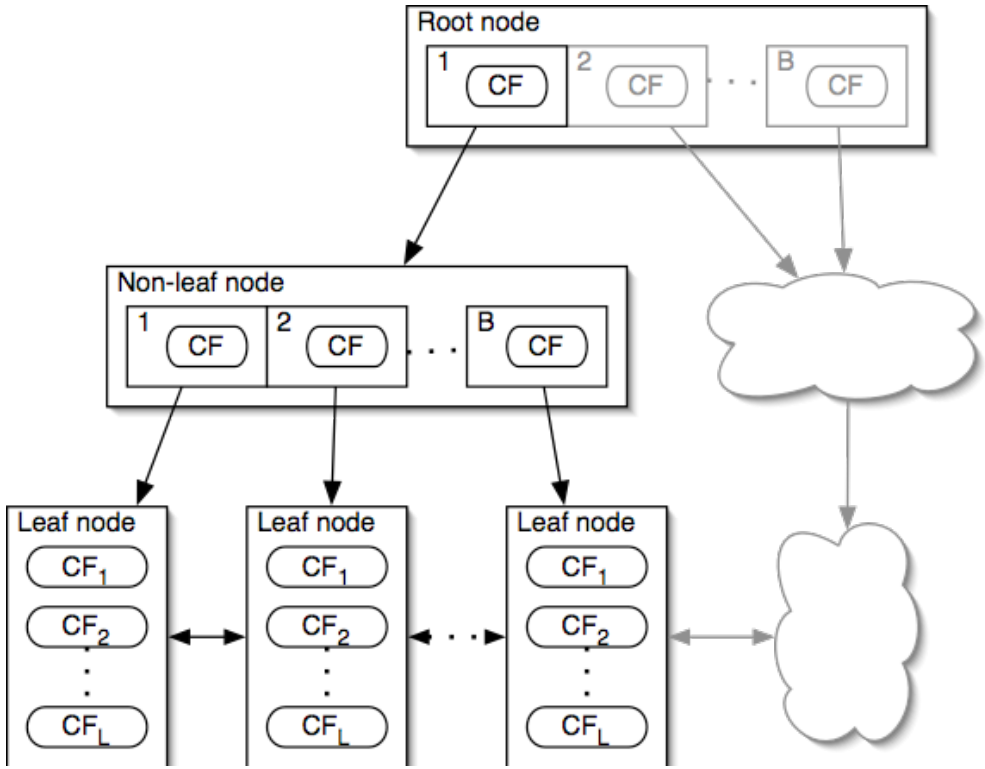
- **CF** entry is a **summary** of statistics of the cluster
- A **representation** of the cluster
- A CF entry has **sufficient information** to calculate the centroid, radius, diameter and many other distance measures
- **Additivity** theorem allows us to **merge sub-clusters incrementally**

CF-tree

B = Branching Factor,
maximum children
in a non-leaf node

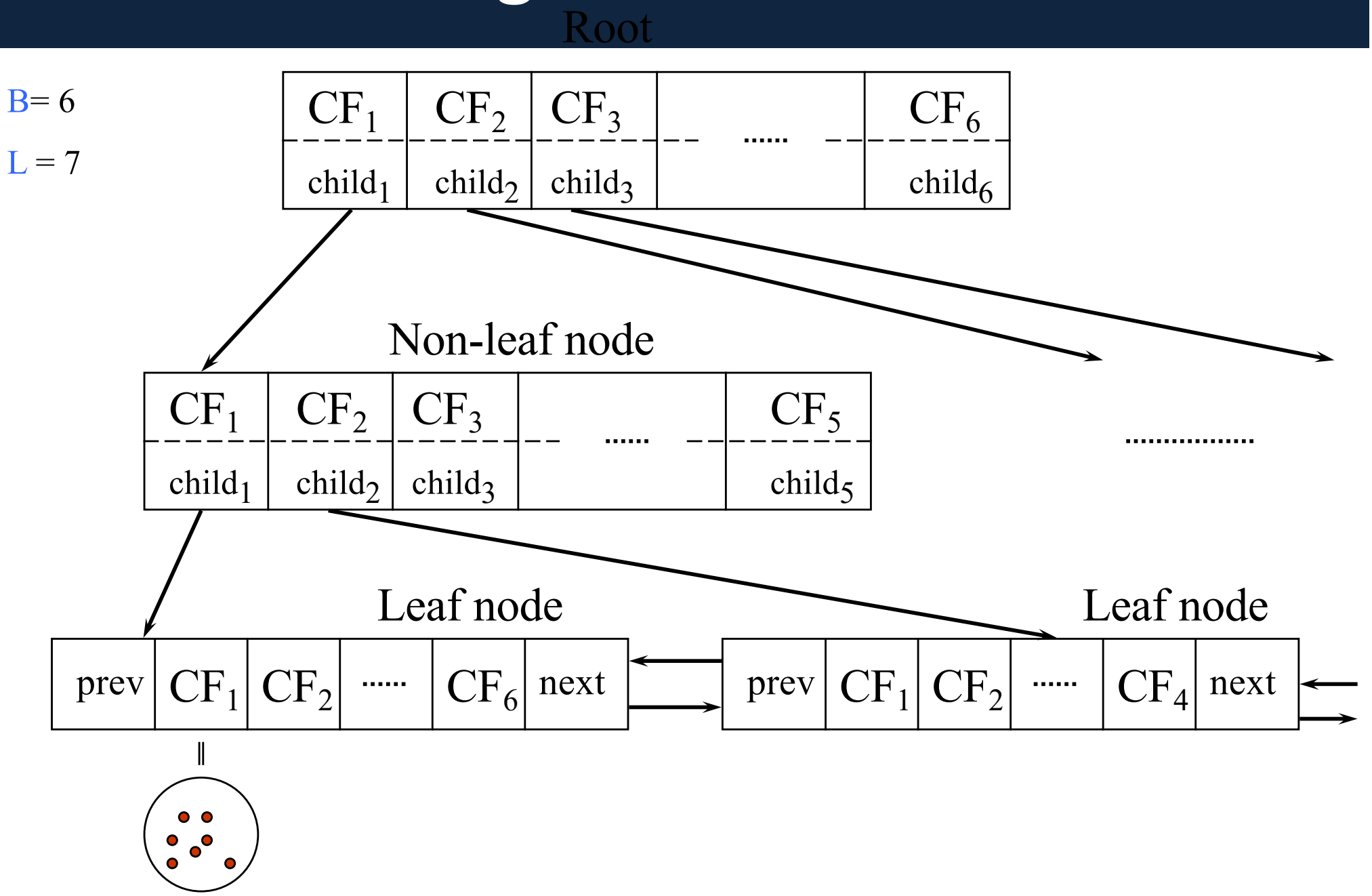
T = Threshold for
diameter or radius
of the cluster in a leaf

L = Each leaf node has at most L CF entries which each satisfy threshold T (e.g., the diameter or radius has to be less than T)



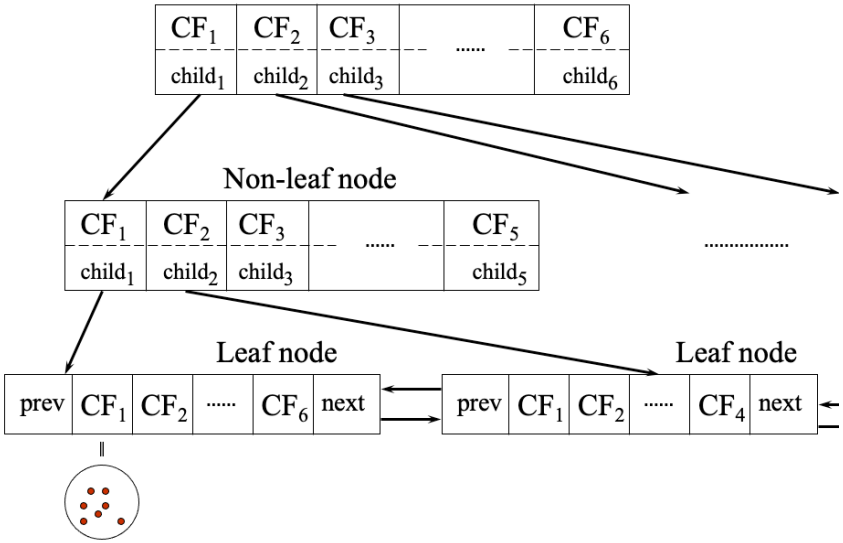
Node size is determined by dimensionality of data points and input parameter P (page size)

The algorithm: CF-tree

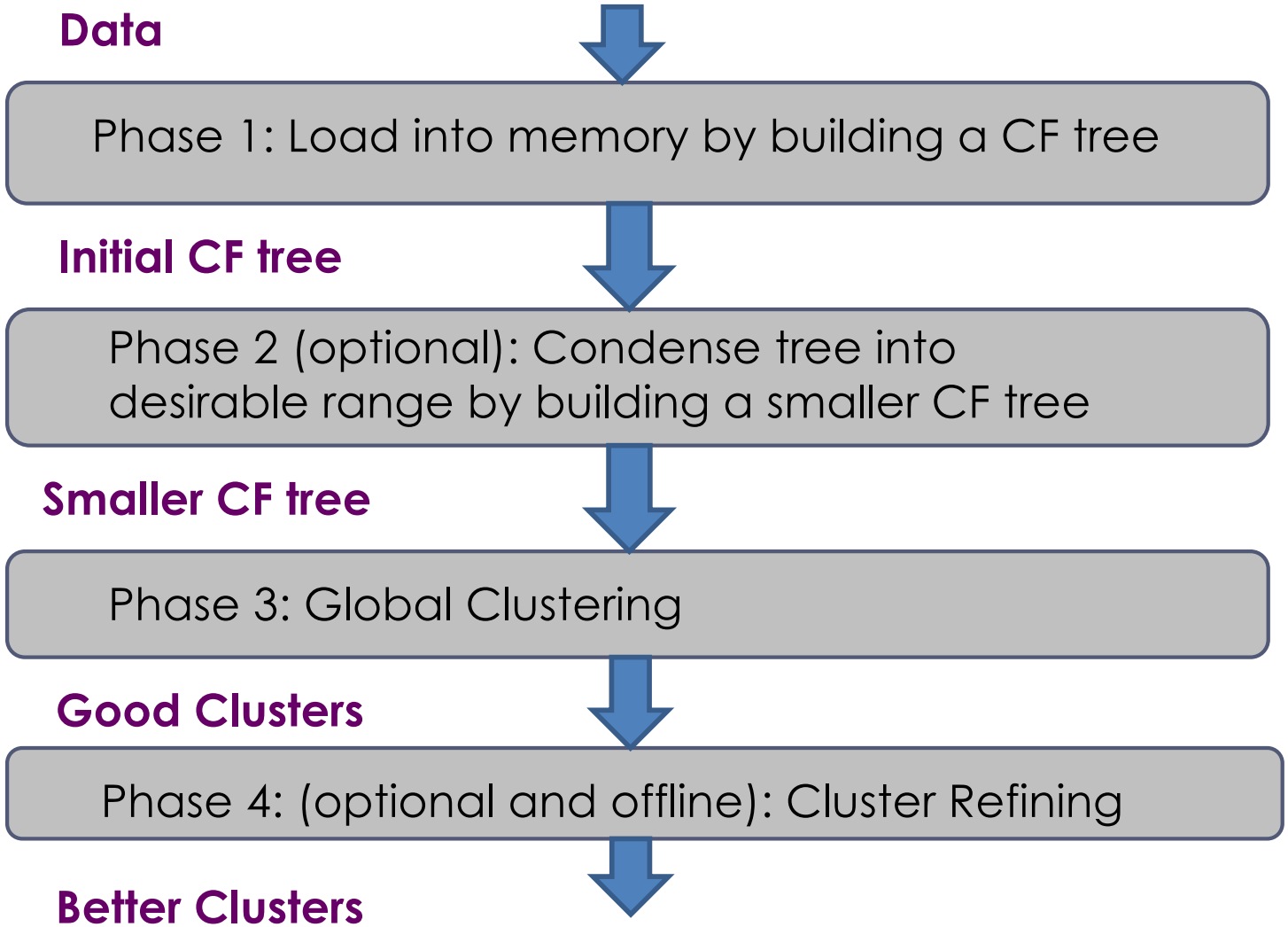


CF tree building process

- Start with the root
- Find the CF entry in the root closest to the data point, move to that child and repeat the process until a closest leaf entry is found.
- At the leaf
  - If the point can be accommodated (absorbed) in the cluster, update the entry
  - If this addition violates the threshold T, split the entry, if this violates the limit imposed by L, split the leaf. If its parent node too is full, split that and so on
- Update the CF entries from the root to the leaf to accommodate this point

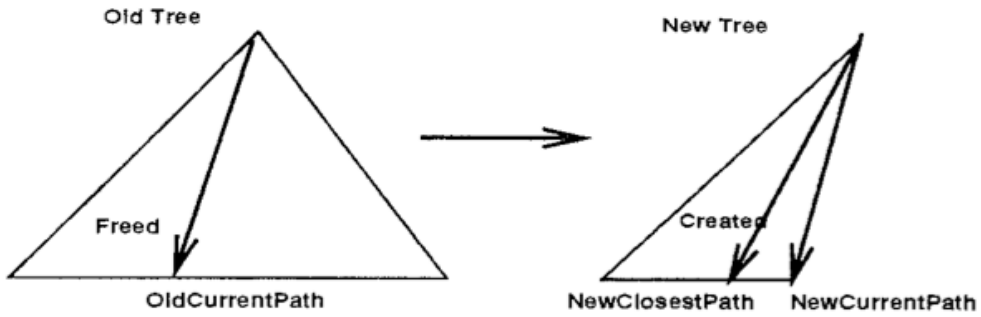


Birch Algorithm



Birch Algorithm: Phase 1

- Choose an initial value for threshold, start inserting the data points one by one into the tree as per the insertion algorithm
- If, in the middle of the above step, the size of the CF tree exceeds the size of the available memory, increase the value of threshold
- Convert the partially built tree into a new tree



- Repeat the above steps until the entire dataset is scanned and a full tree is built
- Outlier Handling

CF-Tree

- The tree size is a function of T
  - The larger T is, the smaller the tree is
- A node needs to fit in a page of size P.
- Once the dimension of the data space is given, the sizes of leaf and non leaf entries are known, the B and L are determined by P. So Page size can be varied for performance turning.

This page size is not really the virtual memory page size. It is a user defined input parameter.

Birch Algorithm: Phase 2,3, and 4

- **Phase 2**
  - A bridge between phase 1 and phase 3
  - Builds a smaller CF tree by increasing the threshold
    - *Use existing clustering algorithm on CF entries*
    - *Helps fix problem where natural clusters span nodes*
- **Phase 3**
  - Apply global clustering algorithm to the sub-clusters given by leaf entries of the CF tree
  - Improves clustering quality
- **Phase 4**
  - Scan the entire dataset to label the data points
  - Outlier handling

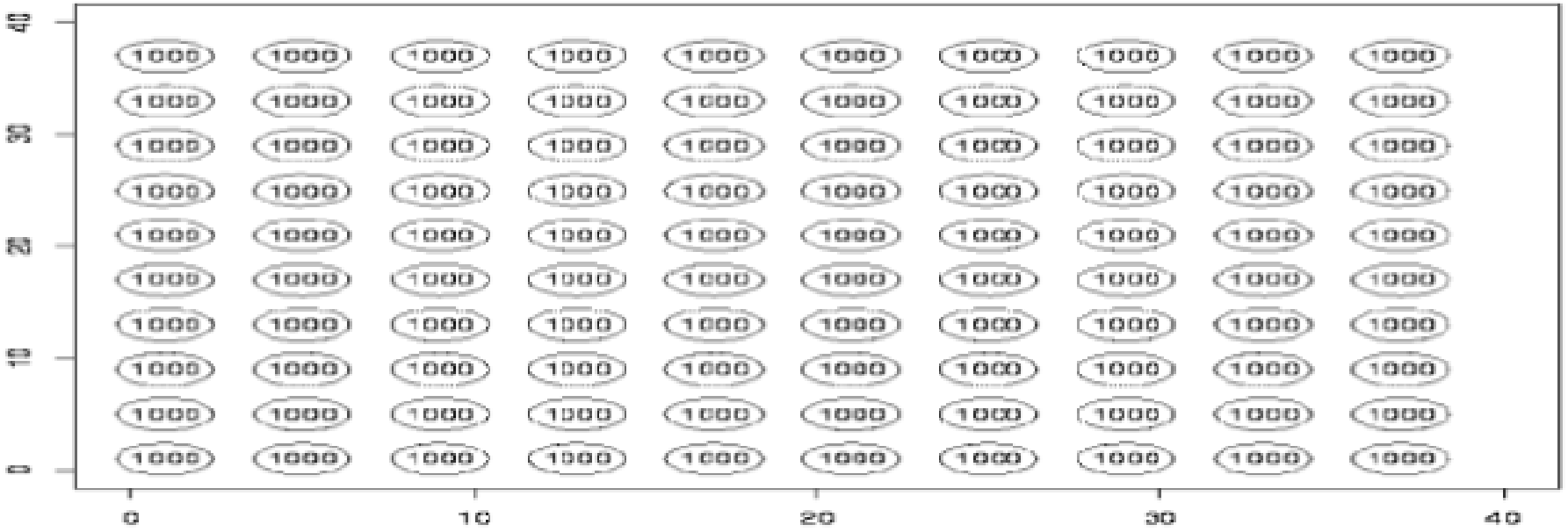
Experiments

- Input parameters:
  - *Memory (*M*): 5% of data set*
  - *Disk space (*R*): 20% of* M
  - *Distance equation:* D2
  - *Quality equation: weighted average diameter (*D*)*
  - *Initial threshold (*T*): 0.0*
  - *Page size (*P*): 1024 bytes*

Experiments

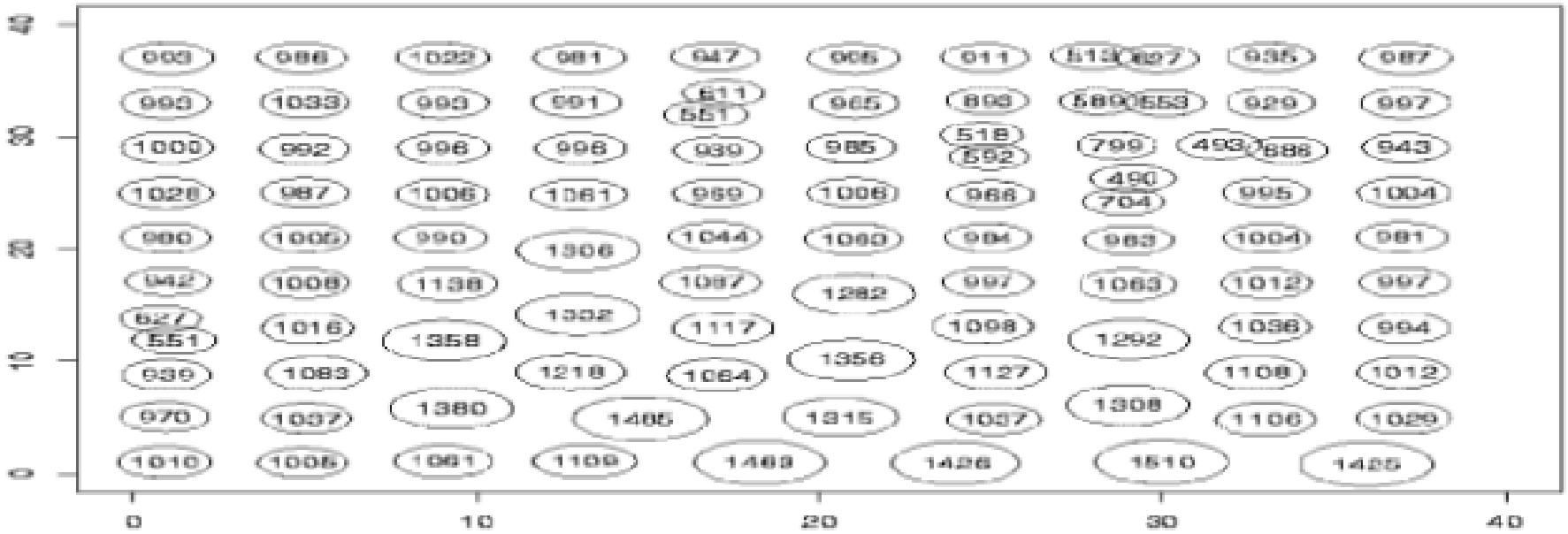
- Create 3 synthetic data sets for testing
  - *Also create an ordered copy for testing input order*
- KMEANS and CLARANS require entire data set to be in memory
  - *Initial scan is from disk, subsequent scans are in memory*

Intended clustering



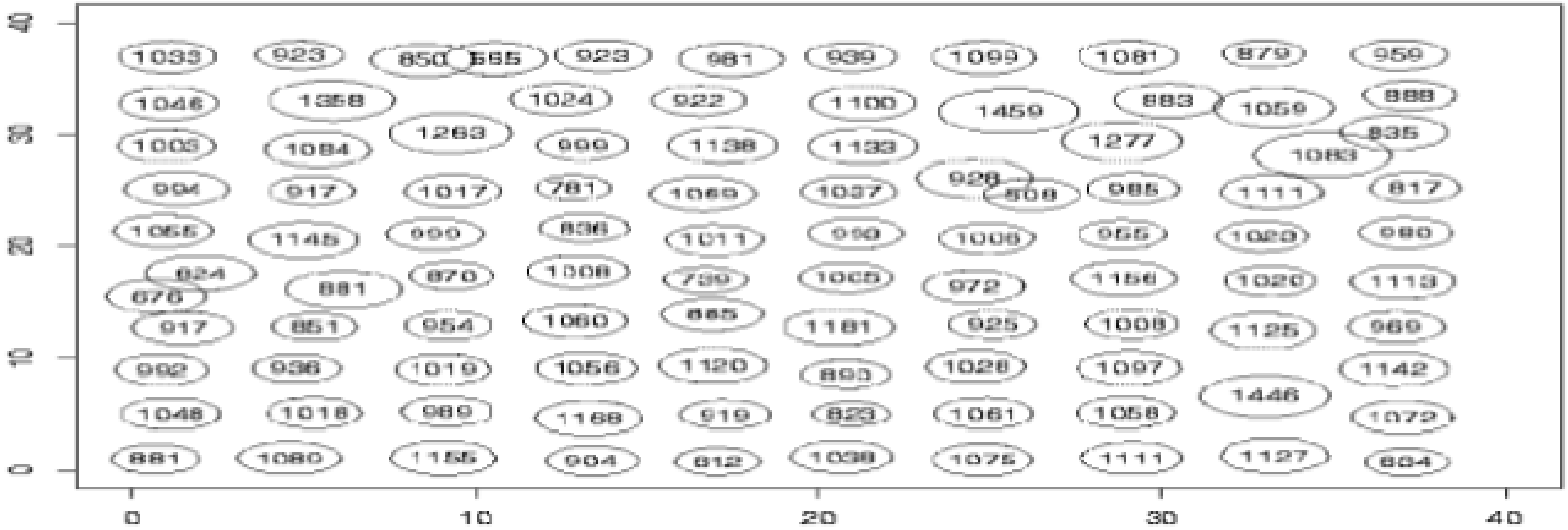
Dataset	Generator Setting	D_{act}
DS1	grid, $K = 100, n_l = n_h = 1000, r_l = r_h = \sqrt{2}, k_g = 4, r_n = 0\%, o = randomized$	2.00
DS2	sine, $K = 100, n_l = n_h = 1000, r_l = r_h = \sqrt{2}, n_c = 4, r_n = 0\%, o = randomized$	2.00
DS3	random, $K = 100, n_l = 0, n_h = 2000, r_l = 0, r_h = 4, r_n = r_n = 0\%, o = randomized$	4.18

Kmeans Clustering



DS	*Time*	*D*	*DS*	*Time*	*D*
1	**43.9**	**2.09**	1o	33.8	1.97
2	13.2	4.43	2o	12.7	4.20
3	32.9	3.66	3o	36.0	4.35

CLARANS clustering

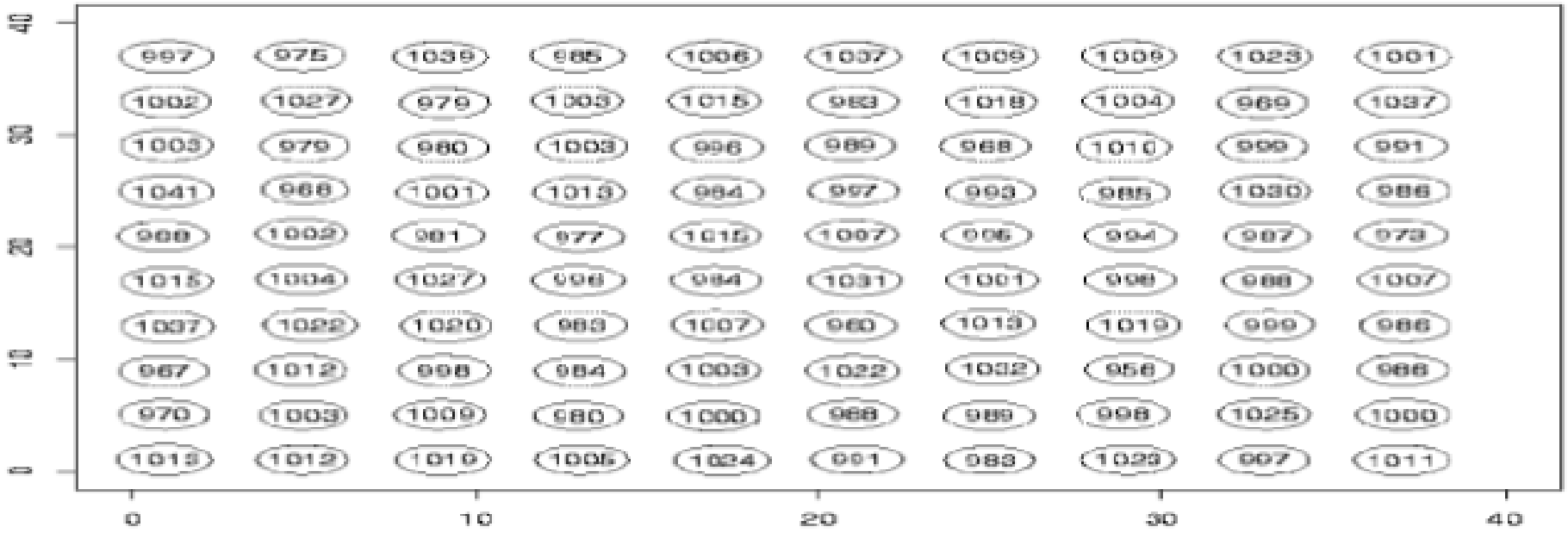


DS	*Time*	*D*	*DS*	*Time*	*D*
1	**932**	**2.11**	1o	1525.7	10.75
2	758	2.56	2o	1495.8	179.23
3	835	3.36	3o	2390.5	6.93

Clustering Results

- The number of data points in both KMeans and CLARANS cluster can be $> 50\%$ different from the number in the actual cluster;
- The pattern of the location of the CLARANS cluster centers is distorted;
- The radii of CLARANS clusters varies largely;
- KMeans is much more efficient in time then CLARANS.

BIRCH Clustering



DS	*Time*	*D*	*DS*	*Time*	*D*
1	**47.1**	**1.87**	1o	47.4	1.87
2	47.5	1.99	2o	46.4	1.99
3	49.5	3.39	3o	48.4	3.26

BIRCH Clustering Results

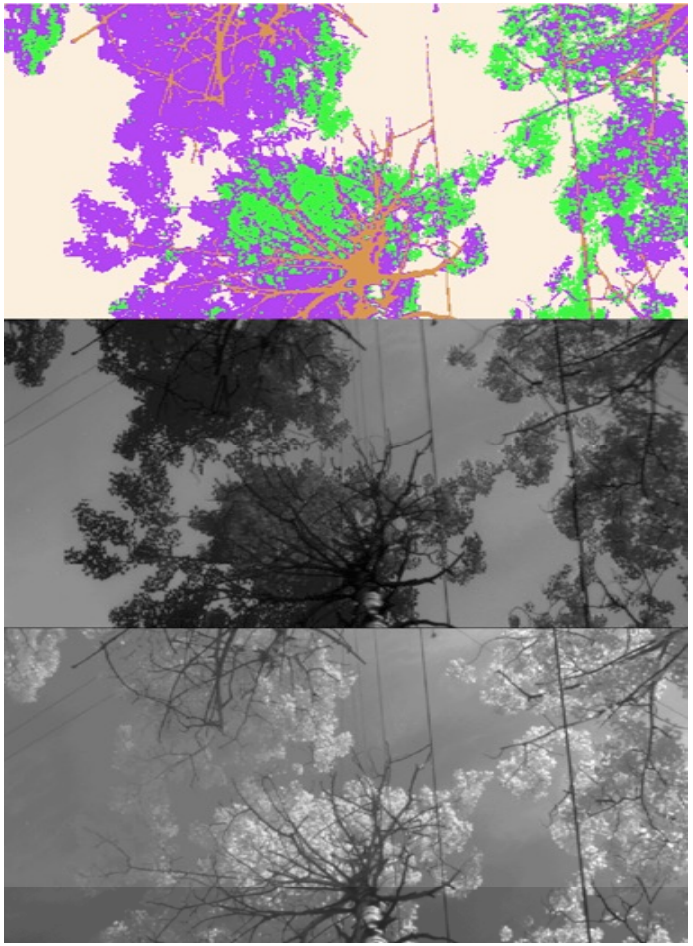
- The BIRCH clusters are very similar to the actual clusters in terms of location, number of points, and radii.
  - The maximal and average difference between the centroids of an actual cluster and its corresponding BIRCH cluster are 0.17 and 0.07, respectively
  - The number of points in a BIRCH cluster is no more than 4% different from the corresponding actual cluster.
  - The radii of the BIRCH clusters (from 1.25 to 1.4) are close those of the actual clusters (1.41)

Compare CLARANS and BIRCH

- CLARANS is at least 15 times slower than BIRCH, and is sensitive to the pattern of the dataset,
- The D value for the CLARANS clusters is much larger than that for the BIRCH clusters
- CLARANS is sensitive to the order of the data appearing in the dataset, while BIRCH's result is not affected by the data ordering

Pixel classification in images

- From top to bottom:
  - Birch Clustering -- First clustering, 5 clusters:
    1. Very bright part of sky
    2. ordinary part of sky
    3. clouds
    4. sunlit leaves
    5. tree branches and shadows on the trees.
  - Second clustering applied to cluster 5 data: Branches and shadows separated apart (NIR weighted 10 times heavier than VIS)
    - *Visible wavelength band*
    - *Near-infrared band*



- BIRCH works with very large data sets
- BIRCH performs faster then CLARANS, while getting better compression and nearly as good quality
- Explicitly bounded by computational resources
  - *Runs with specified amount of memory (*P*)*
- Superior to CLARANS and KMEANS
  - *Quality, speed, stability and scalability*

Clustering Evaluation

- Internal measures
- Stability measures

Internal Measures

- Connectivity measures the degree of connectedness of the clusters, as determined by the *k*-nearest neighbors.
- The Dunn index is the ratio between the smallest distances between observations not in the same cluster to the largest intra-cluster distance.
- The Silhouette width value measures the degree of confidence in a particular clustering assignment.

Stability Measures

- To measure the stability of the generated clusters, clustering was applied repeatedly to noisy data produced by removing one column at a time from the original data.
- The average difference between the clusters generated from the noisy data and the original data was computed.

Stability Measures

- Non-overlapping (APN) measure,
- the Average Distance (AD) measure,
- the Average Distance between Means (ADM) measure, and
- the Figure of Merit (FOM) measure.

Stability Measures

- The APN measures the average proportion of observations not placed in the same cluster by clustering based on the full data and clustering based on the noisy data.
- The AD measure computes the average distance between observations placed in the same cluster by clustering based on the full data and clustering based on the noisy data.

Stability Measures

- The ADM measure computes the average distance between cluster centers for observations placed in the same cluster by clustering based on the full data and clustering based on the noisy data.
- The FOM measures the average intra-cluster variance of the observations in the designated deleted column, where the clustering is based on the remaining (undeleted) samples