
CS481: Bioinformatics Algorithms

Can Alkan

EA224

`calkan@cs.bilkent.edu.tr`

<http://www.cs.bilkent.edu.tr/~calkan/teaching/cs481/>

QUIZ 3

Quiz 3

Show the phylogenetic tree construction using UPGMA for the following score matrix:

	A	B	C	D	E	F
A						
B	19					
C	27	31				
D	8	18	26			
E	33	36	41	31		
F	18	1	32	17	35	

Quiz 3

	A	B	C	D	E	F
A						
B	19					
C	27	31				
D	8	18	26			
E	33	36	41	31		
F	18	1	32	17	35	

$$h_{BF} = \frac{1}{2} = 0.5$$

$$C = \{A, C, D, E, (B, F)\}$$

$$D_{BF, A} = \frac{1}{|\{B, F\}| * |\{A\}|} D_{B, A} + \mathcal{D}_{F, A}$$
$$= \frac{1}{2} (19 + 8) = 8.5$$

	A	BF	C	D	E
A					
BF	18.5				
C	27	31.5			
D	8	17.5	26		
E	33	35.5	41	31	

Quiz 3

$$C = \{(A,D), C, E, (B,F)\}$$

	A	BF	C	D	E
A					
BF	18.5				
C	27	31.5			
D	8	17.5	26		
E	33	35.5	41	31	

$$D_{BF, A} = \frac{D_{B, A} + \mathcal{D}_{F, A} + D_{B, D} + \mathcal{D}_{F, D}}{|\{B, F\}| * |\{A, D\}|}$$

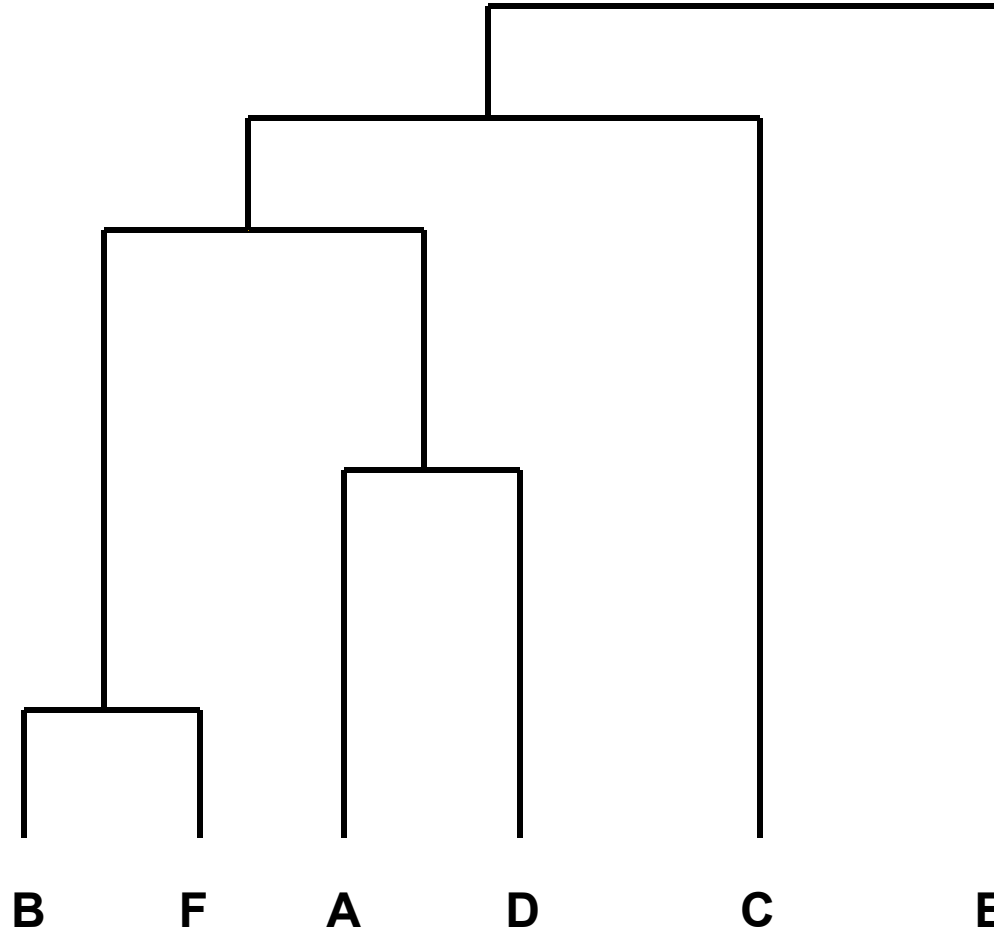
$$= \frac{(19 + 8 + 8 + 7)}{4} = 8$$

$$h_{BF} = \frac{1}{2} = 0.5$$

$$h_{AD} = \frac{8}{2} = 4$$

	AD	BF	C	E
AD				
BF	18			
C	26.5	31.5		
E	32	35.5	41	

Quiz 3



CLUSTERING

Applications of Clustering

- Viewing and analyzing vast amounts of biological data as a whole set can be infeasible
 - It is easier to interpret the data if they are partitioned into clusters combining similar data points.
-

Inferring Gene Functionality

- Researchers want to know the functions of newly sequenced genes
 - Simply comparing the new gene sequences to known DNA sequences often does not give away the function of gene
 - For 40% of sequenced genes, functionality cannot be ascertained by only comparing to sequences of other known genes
 - Microarrays allow biologists to infer gene function even when sequence similarity alone is insufficient to infer function.
-

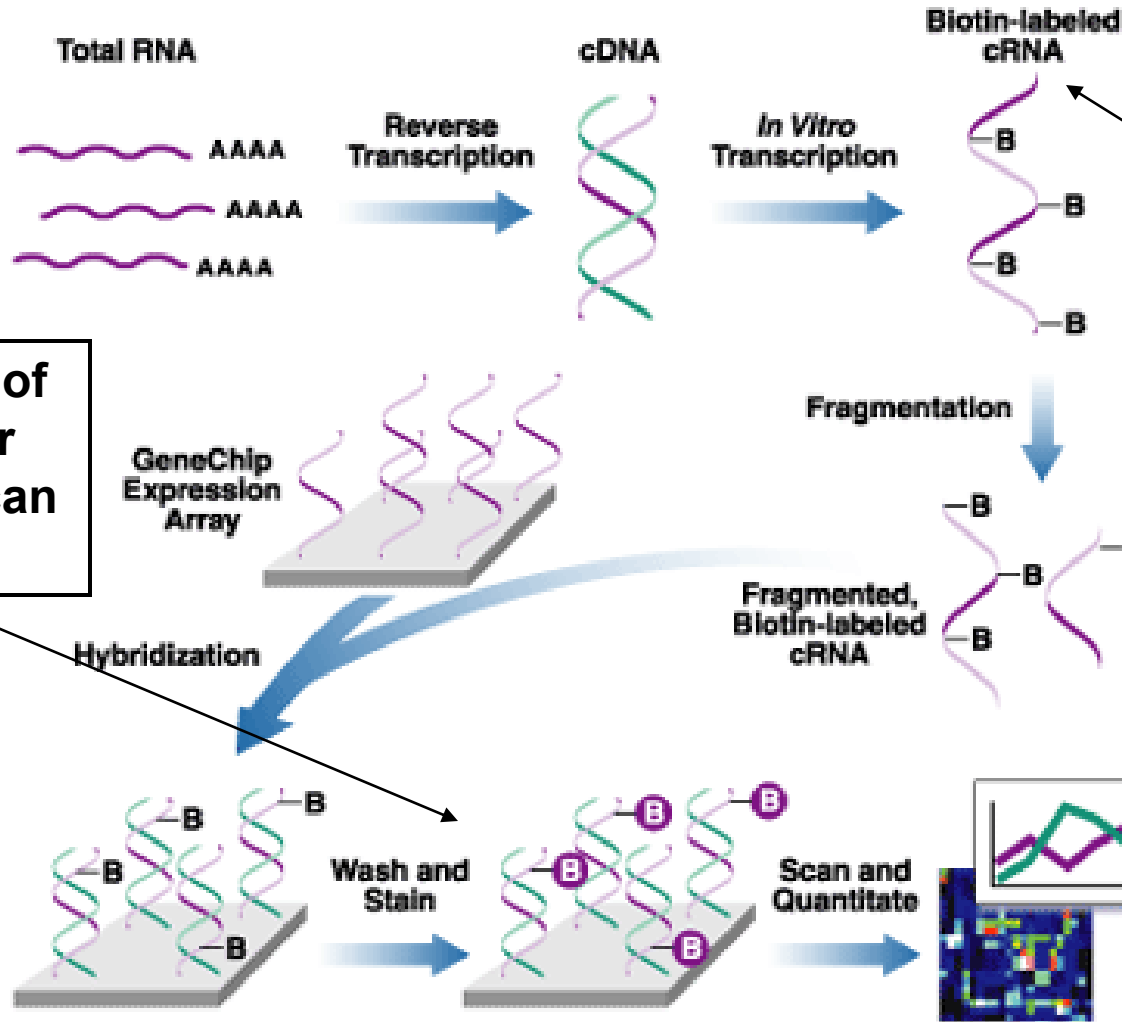
Microarrays and Expression Analysis

- Microarrays measure the activity (expression level) of the genes under varying conditions/time points
 - Expression level is estimated by measuring the amount of mRNA for that particular gene
 - A gene is active if it is being transcribed
 - More mRNA usually indicates more gene activity
-

Microarray Experiments

- ❑ Produce cDNA from mRNA (DNA is more stable)
 - ❑ Attach phosphor to cDNA to see when a particular gene is expressed
 - ❑ Different color phosphors are available to compare many samples at once
 - ❑ Hybridize cDNA over the micro array
 - ❑ Scan the microarray with a phosphor-illuminating laser
 - ❑ Illumination reveals transcribed genes
 - ❑ Scan microarray multiple times for the different color phosphor's
-

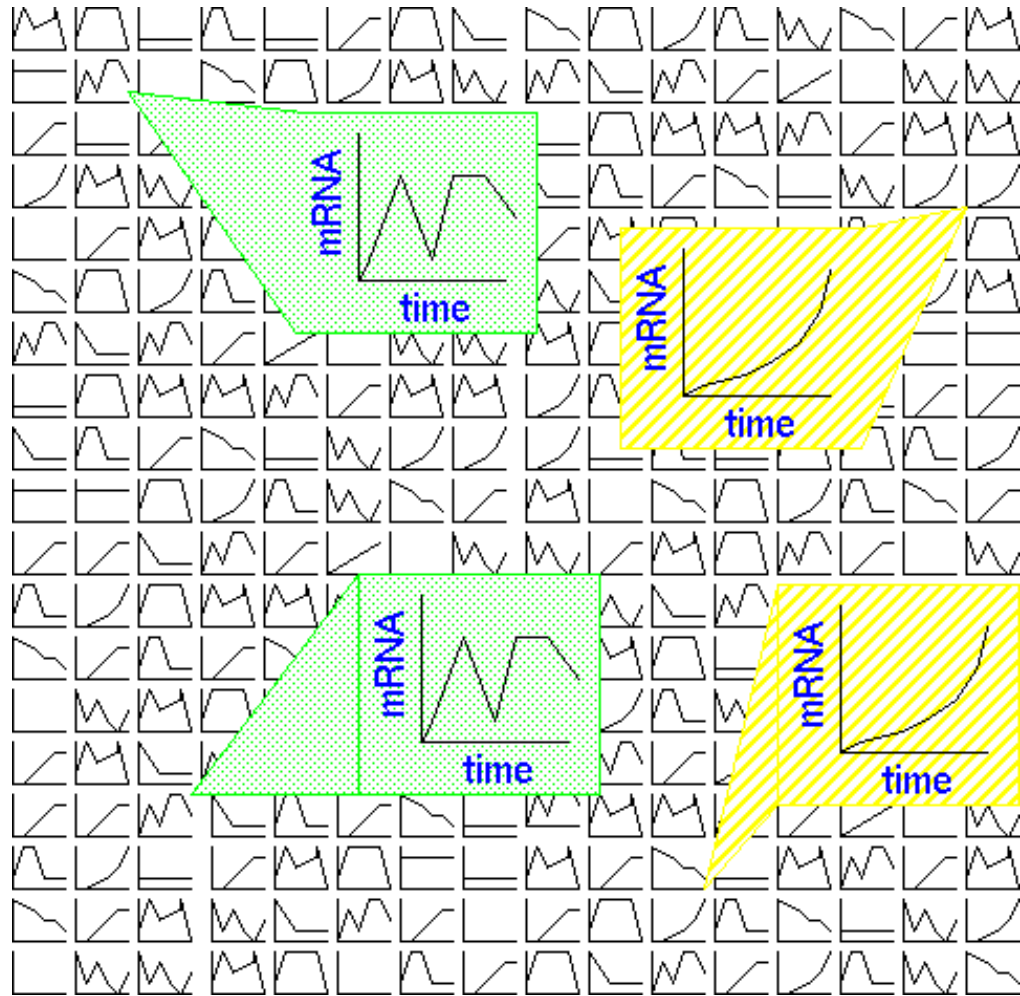
Microarray Experiments (con't)



Then instead of staining, laser illumination can be used

Phosphors can be added here instead

Using Microarrays

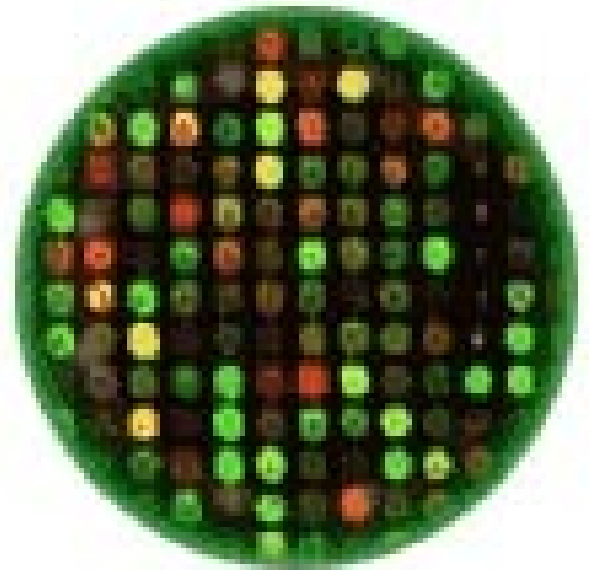


- Track the sample over a period of time to see gene expression over time
- Track two different samples under the same conditions to see the difference in gene expressions

Each box represents one gene's expression over time

Using Microarrays (cont'd)

- **Green:** expressed only from control
- **Red:** expressed only from experimental cell
- **Yellow:** equally expressed in both samples
- **Black:** NOT expressed in either control or experimental cells



Microarray Data

- Microarray data are usually transformed into an **intensity matrix** (below)
- The intensity matrix allows biologists to make correlations between different genes (even if they are dissimilar) and to understand how genes functions might be related

**Intensity (expression level)
of gene at measured time**



Time:	Time X	Time Y	Time Z
Gene 1	10	8	10
Gene 2	10	0	9
Gene 3	4	8.6	3
Gene 4	7	8	3
Gene 5	1	2	3

Clustering of Microarray Data

- Plot each datum as a point in N-dimensional space
 - Make a distance matrix for the distance between every two gene points in the N-dimensional space
 - Genes with a small distance share the same expression characteristics and might be functionally related or similar.
 - Clustering reveal groups of functionally related genes
-

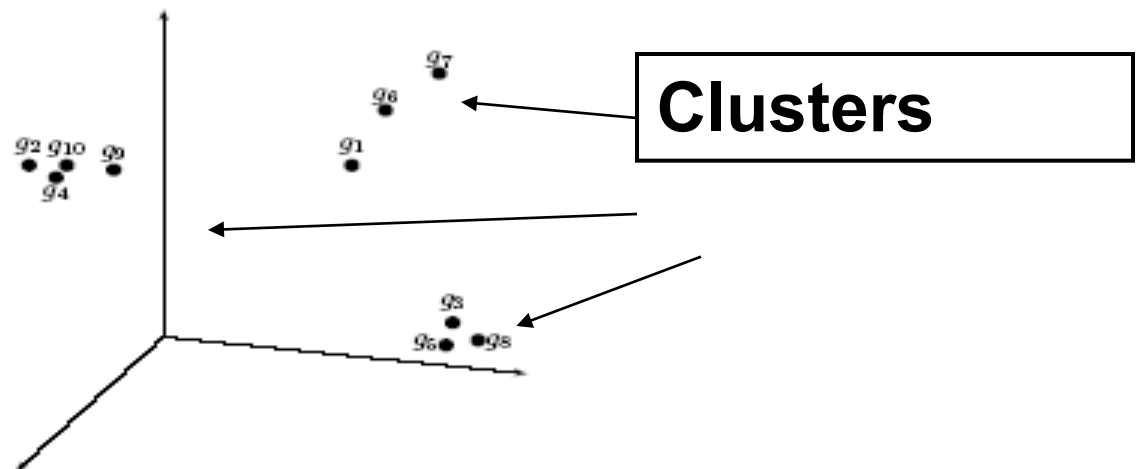
Clustering of Microarray Data (cont'd)

Time	1 hr	2 hr	3 hr
g_1	10.0	8.0	10.0
g_2	10.0	0.0	9.0
g_3	4.0	8.5	3.0
g_4	9.5	0.5	8.5
g_5	4.5	8.5	2.5
g_6	10.5	9.0	12.0
g_7	5.0	8.5	11.0
g_8	2.7	8.7	2.0
g_9	9.7	2.0	9.0
g_{10}	10.2	1.0	9.2

(a) Intensity matrix, I

	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}
g_1	0.0	8.1	9.2	7.7	9.3	2.3	5.1	10.2	6.1	7.0
g_2	8.1	0.0	12.0	0.9	12.0	9.5	10.1	12.8	2.0	1.0
g_3	9.2	12.0	0.0	11.2	0.7	11.1	8.1	1.1	10.5	11.5
g_4	7.7	0.9	11.2	0.0	11.2	9.2	9.5	12.0	1.6	1.1
g_5	9.3	12.0	0.7	11.2	0.0	11.2	8.5	1.0	10.6	11.6
g_6	2.3	9.5	11.1	9.2	11.2	0.0	5.6	12.1	7.7	8.5
g_7	5.1	10.1	8.1	9.5	8.5	5.6	0.0	9.1	8.3	9.3
g_8	10.2	12.8	1.1	12.0	1.0	12.1	9.1	0.0	11.4	12.4
g_9	6.1	2.0	10.5	1.6	10.6	7.7	8.3	11.4	0.0	1.1
g_{10}	7.0	1.0	11.5	1.1	11.6	8.5	9.3	12.4	1.1	0.0

(b) Distance matrix, d

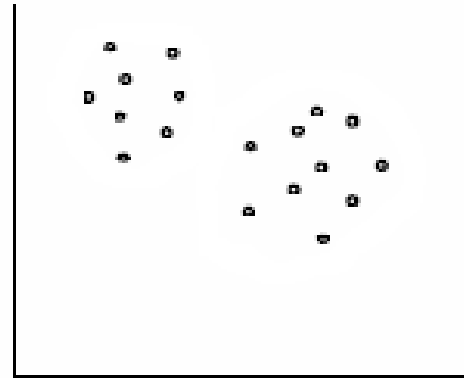


(c) Expression patterns as points in three-dimensional space.

Homogeneity and Separation Principles

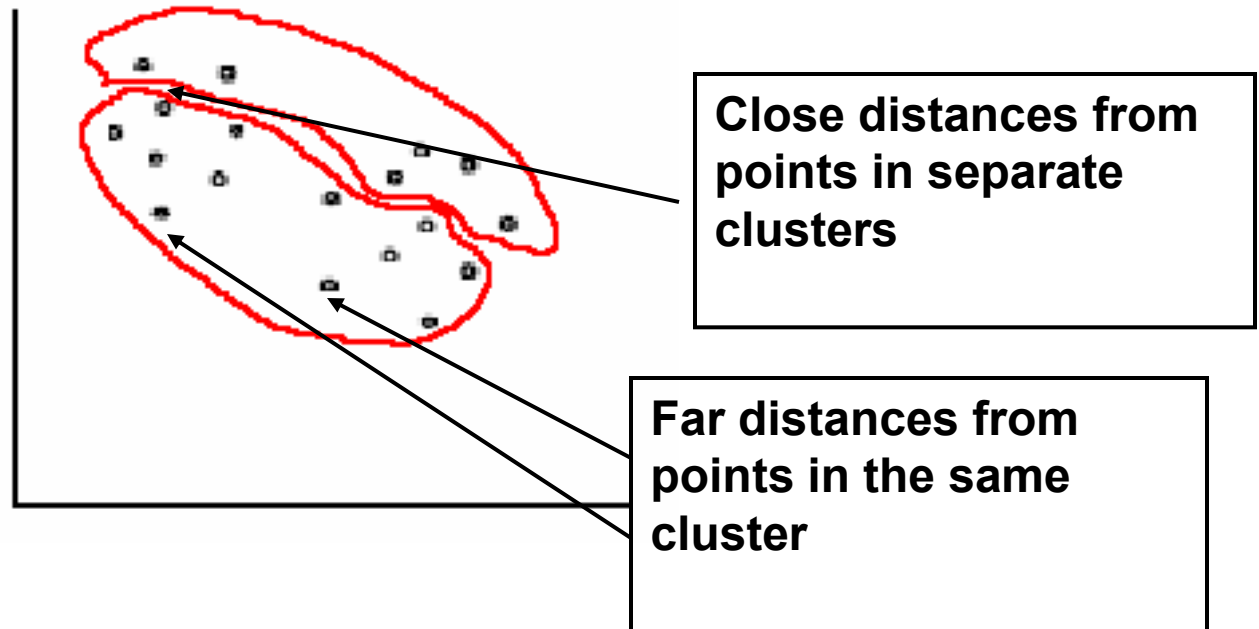
- **Homogeneity:** Elements within a cluster are close to each other
- **Separation:** Elements in different clusters are further apart from each other
- ...clustering is not an easy task!

Given these points a clustering algorithm → might make two distinct clusters as follows



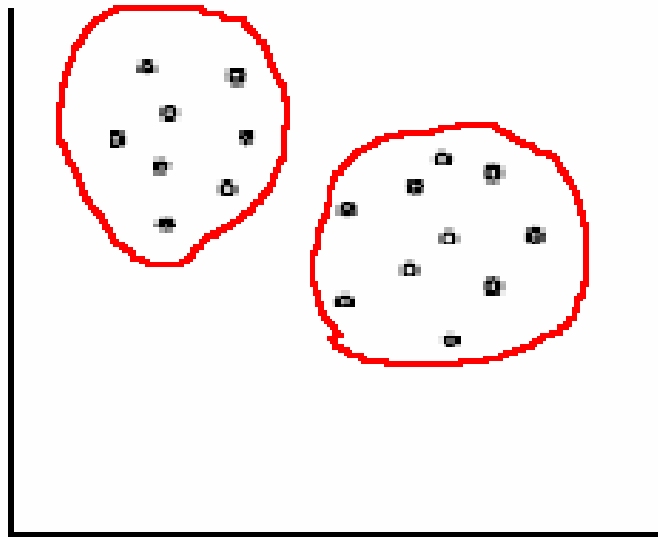
Bad Clustering

This clustering violates both Homogeneity and Separation principles



Good Clustering

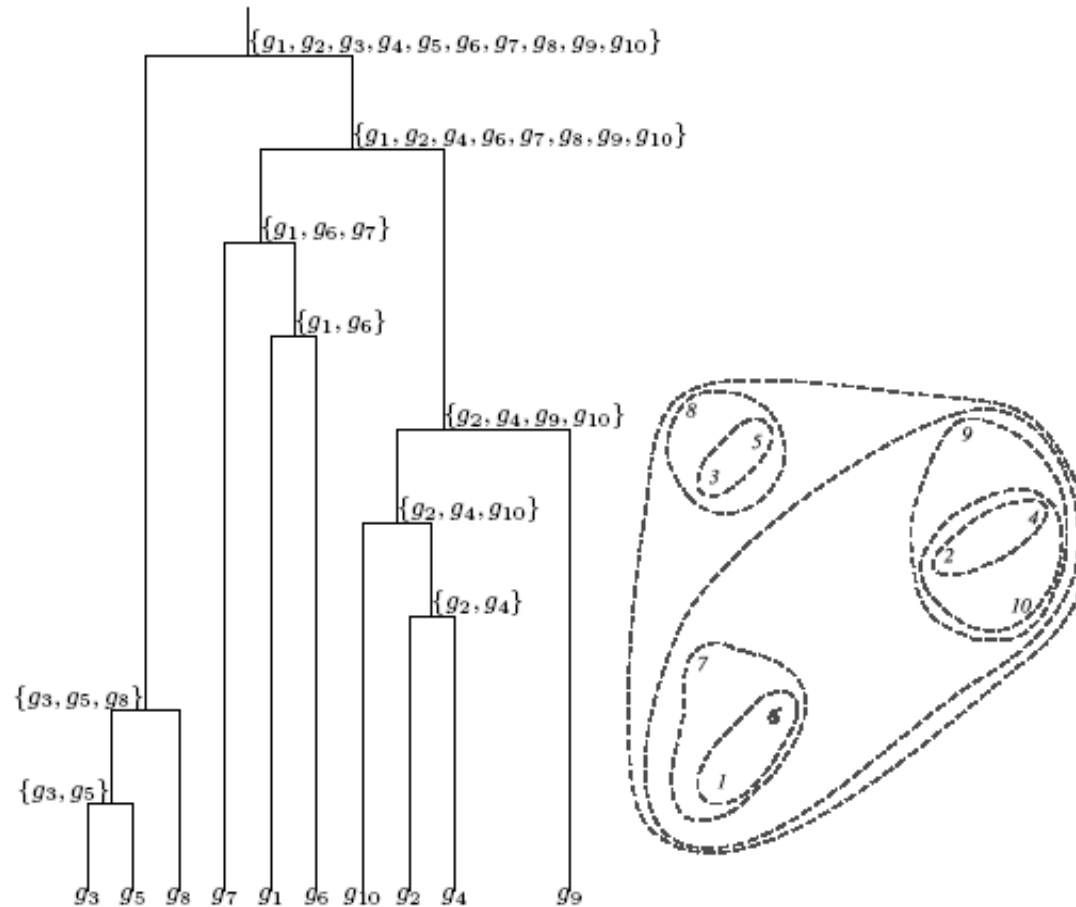
This clustering satisfies both Homogeneity and Separation principles



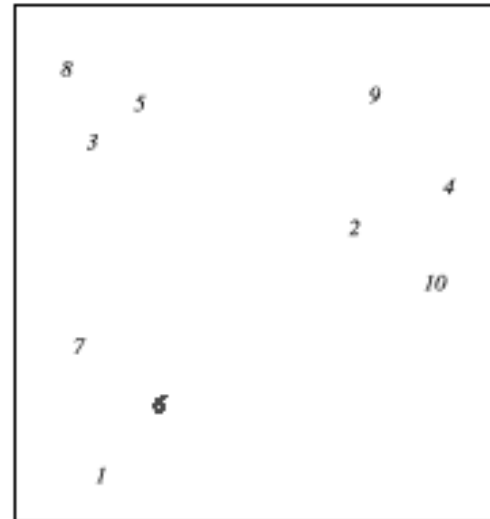
Clustering Techniques

- **Agglomerative:** Start with every element in its own cluster, and iteratively join clusters together
 - **Divisive:** Start with one cluster and iteratively divide it into smaller clusters
 - **Hierarchical:** Organize elements into a tree, leaves represent genes and the length of the pathes between leaves represents the distances between genes. Similar genes lie within the same subtrees
-

Hierarchical Clustering

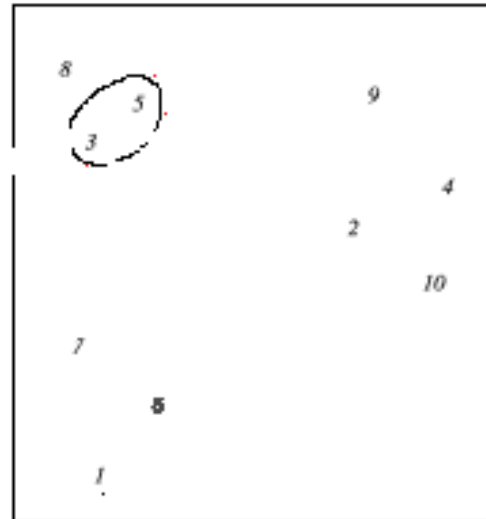
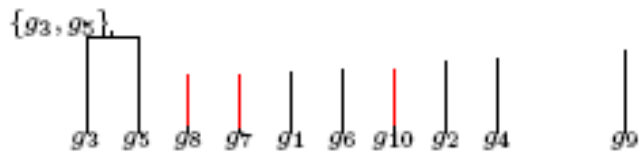


Hierarchical Clustering: Example

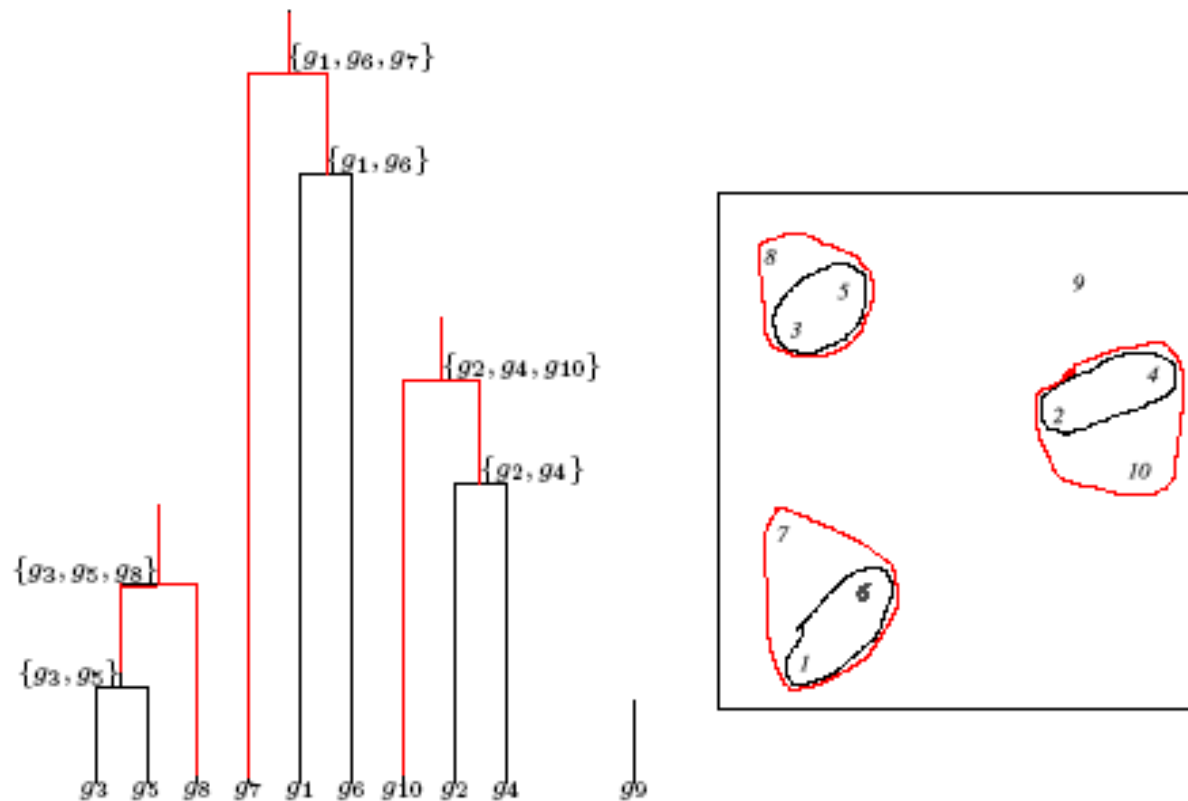


g3 g5 g8 g7 g1 g6 g10 g2 g4 g9

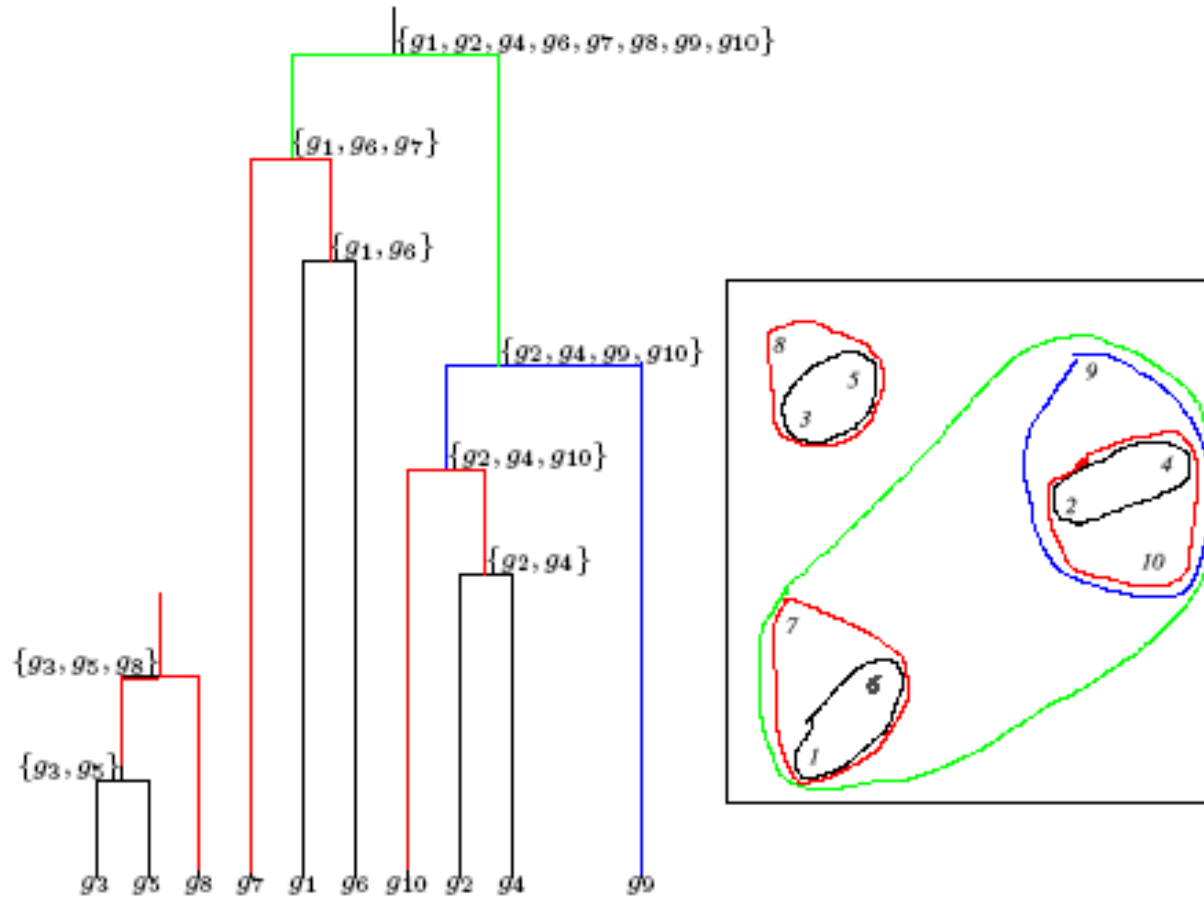
Hierarchical Clustering: Example



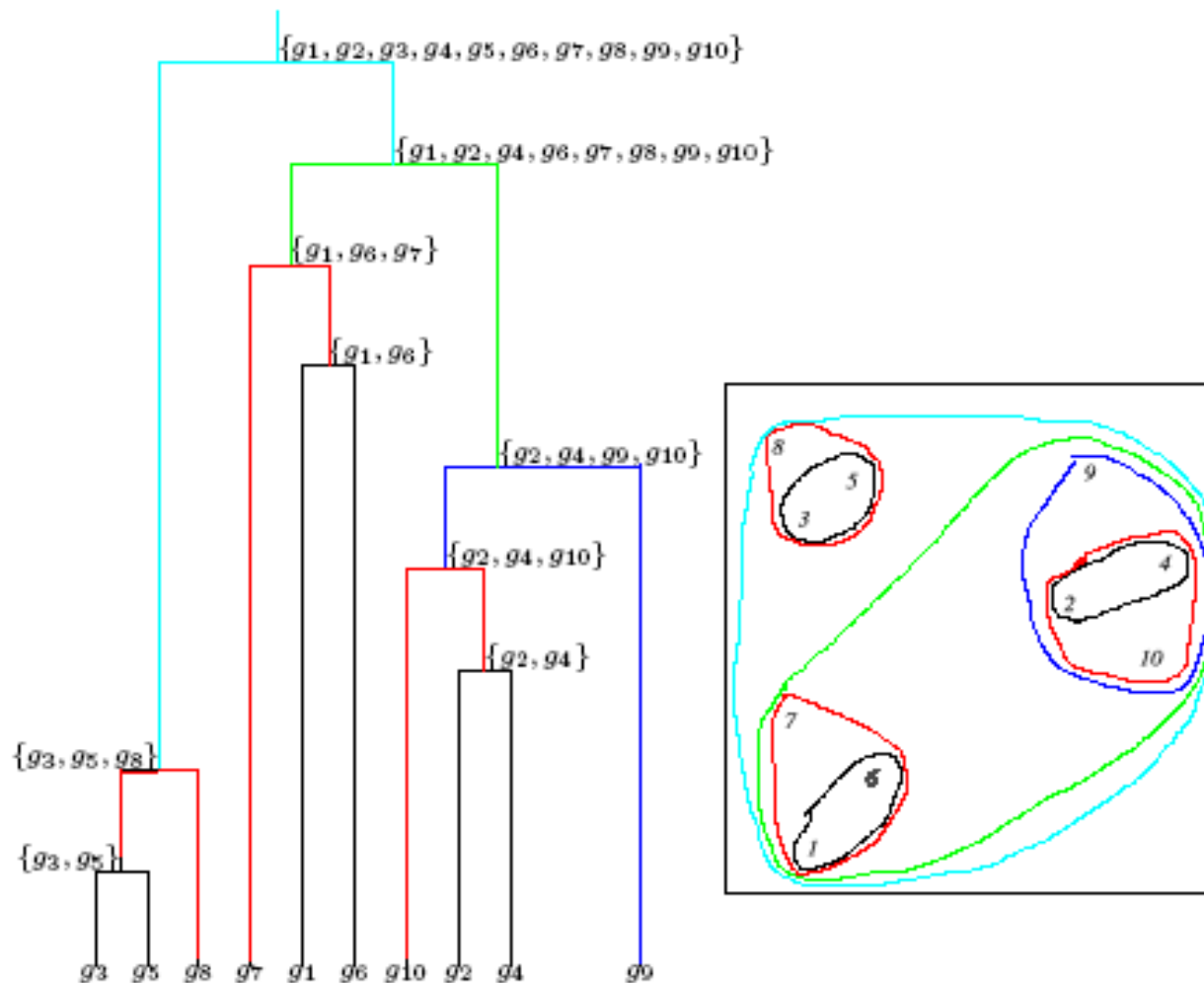
Hierarchical Clustering: Example



Hierarchical Clustering: Example

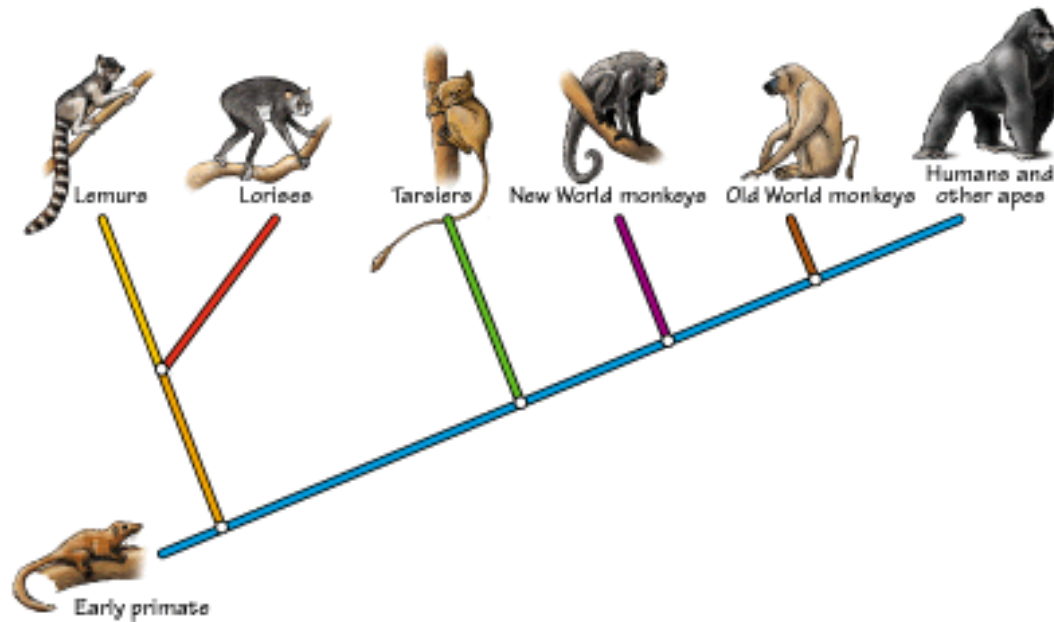


Hierarchical Clustering: Example



Hierarchical Clustering (cont'd)

- Hierarchical Clustering is often used to reveal evolutionary history



Hierarchical Clustering Algorithm

1. Hierarchical Clustering (d, n)
2. Form n clusters each with one element
3. Construct a graph T by assigning one vertex to each cluster
4. **while** there is more than one cluster
5. Find the two closest clusters C_1 and C_2
6. Merge C_1 and C_2 into new cluster C with $|C_1| + |C_2|$ elements
7. **Compute distance from C to all other clusters**
8. Add a new vertex C to T and connect to vertices C_1 and C_2
9. Remove rows and columns of d corresponding to C_1 and C_2
10. Add a row and column to d corresponding to the new cluster C
11. **return T**

The algorithm takes a $n \times n$ distance matrix d of pairwise distances between points as an input.

Hierarchical Clustering Algorithm

1. Hierarchical Clustering (d, n)
2. Form n clusters each with one element
3. Construct a graph T by assigning one vertex to each cluster
4. **while** there is more than one cluster
5. Find the two closest clusters C_1 and C_2
6. Merge C_1 and C_2 into new cluster C with $|C_1| + |C_2|$ elements
7. **Compute distance from C to all other clusters**
8. Add a new vertex C to T and connect to vertices C_1 and C_2
9. Remove rows and columns of d corresponding to C_1 and C_2
10. Add a row and column to d corresponding to the new cluster C
11. return T

Different ways to define distances between clusters may lead to different clusterings

Hierarchical Clustering: Recomputing Distances

- $d_{min}(C, C^*) = \min d(x,y)$
*for all elements x in C and y in C^**
 - Distance between two clusters is the **smallest** distance between any pair of their elements
 - $d_{avg}(C, C^*) = (1 / |C^*||C|) \sum d(x,y)$
*for all elements x in C and y in C^**
 - Distance between two clusters is the **average** distance between all pairs of their elements
-

Squared Error Distortion

- Given a data point \mathbf{v} and a set of points \mathbf{X} , define the **distance** from \mathbf{v} to \mathbf{X}

$$d(\mathbf{v}, \mathbf{X})$$

as the (Euclidian) distance from \mathbf{v} to the *closest* point from \mathbf{X} .

- Given a set of n data points $V = \{v_1 \dots v_n\}$ and a set of k points \mathbf{X} , define the **Squared Error Distortion**

$$d(V, \mathbf{X}) = \sum d(v_i, \mathbf{X})^2 / n \quad 1 \leq i \leq n$$



K-Means Clustering Problem: Formulation

- **Input:** A set, \mathbf{V} , consisting of n points and a parameter k
 - **Output:** A set \mathbf{X} consisting of k points (*cluster centers*) that minimizes the squared error distortion $d(\mathbf{V}, \mathbf{X})$ over all possible choices of \mathbf{X}
-

1-Means Clustering Problem: an Easy Case

- **Input:** A set, \mathbf{V} , consisting of n points
 - **Output:** A **single** point \mathbf{x} (*cluster center*) that minimizes the squared error distortion $d(\mathbf{V}, \mathbf{x})$ over all possible choices of \mathbf{x}
-

1-Means Clustering Problem: an Easy Case

- **Input:** A set, \mathbf{V} , consisting of n points
- **Output:** A **single** point \mathbf{x} (cluster center) that minimizes the squared error distortion $d(\mathbf{V}, \mathbf{x})$ over all possible choices of \mathbf{x}

1-Means Clustering problem is easy.

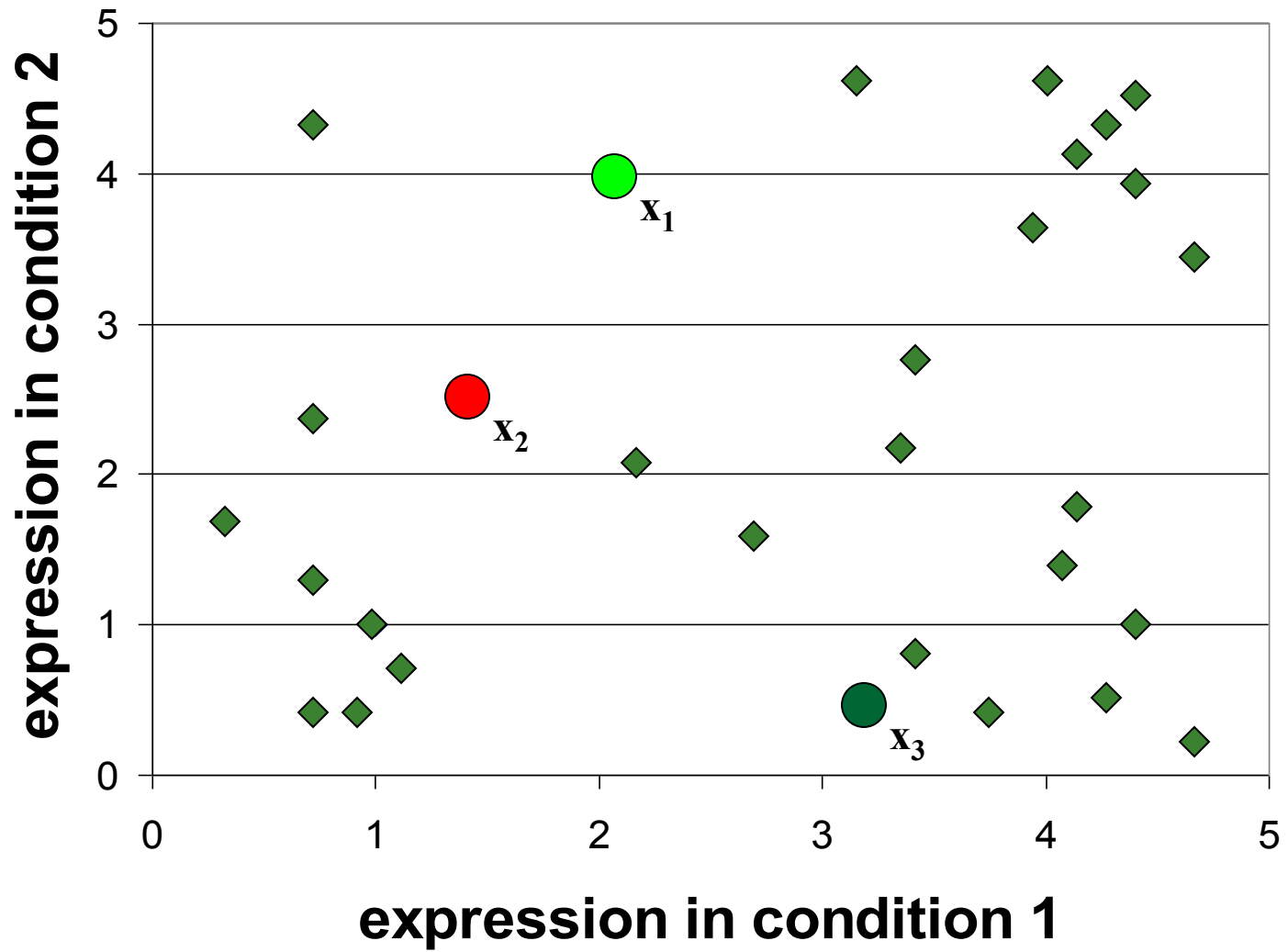
However, it becomes very difficult (NP-complete) for more than one center.

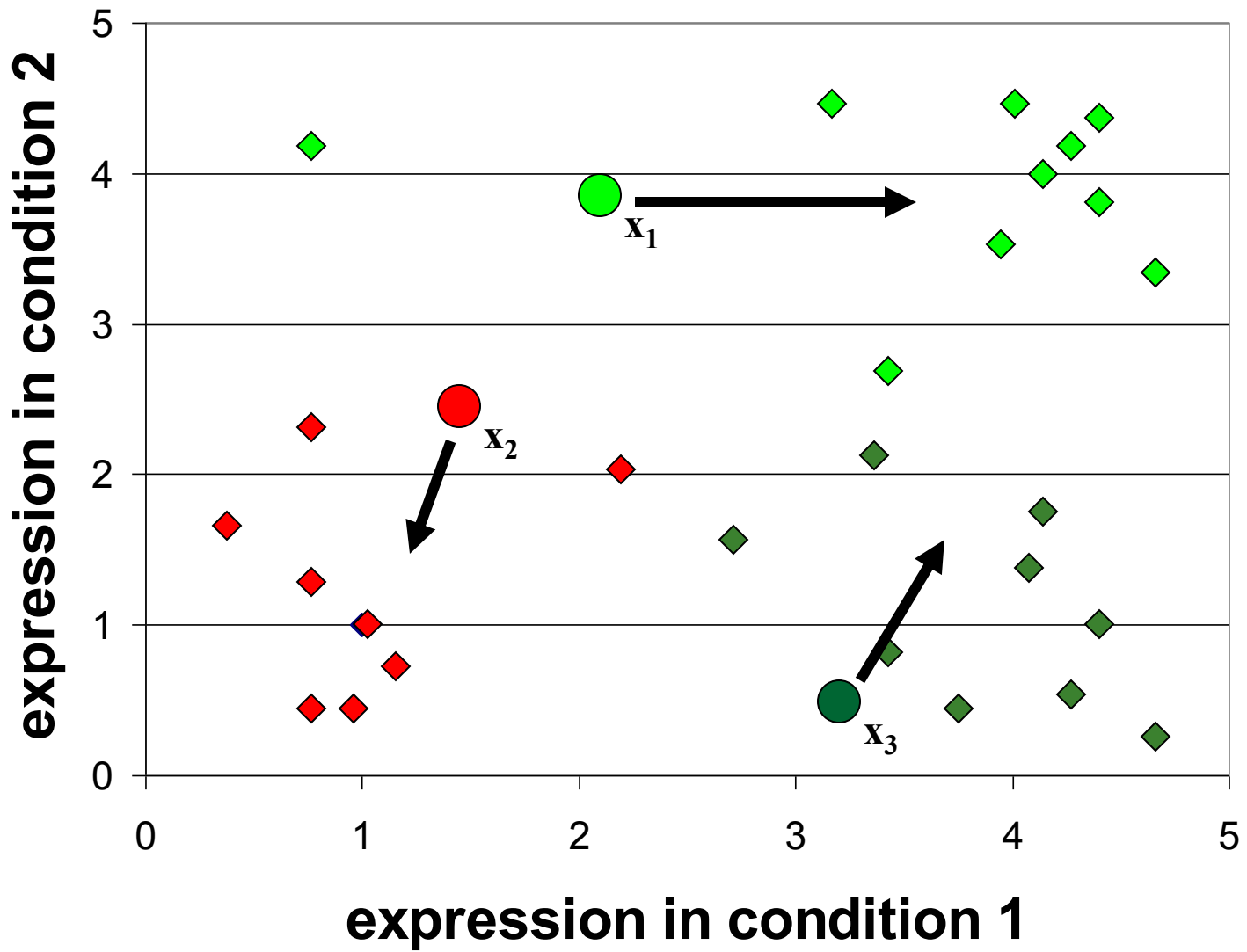
An efficient *heuristic* method for K-Means clustering is the Lloyd algorithm

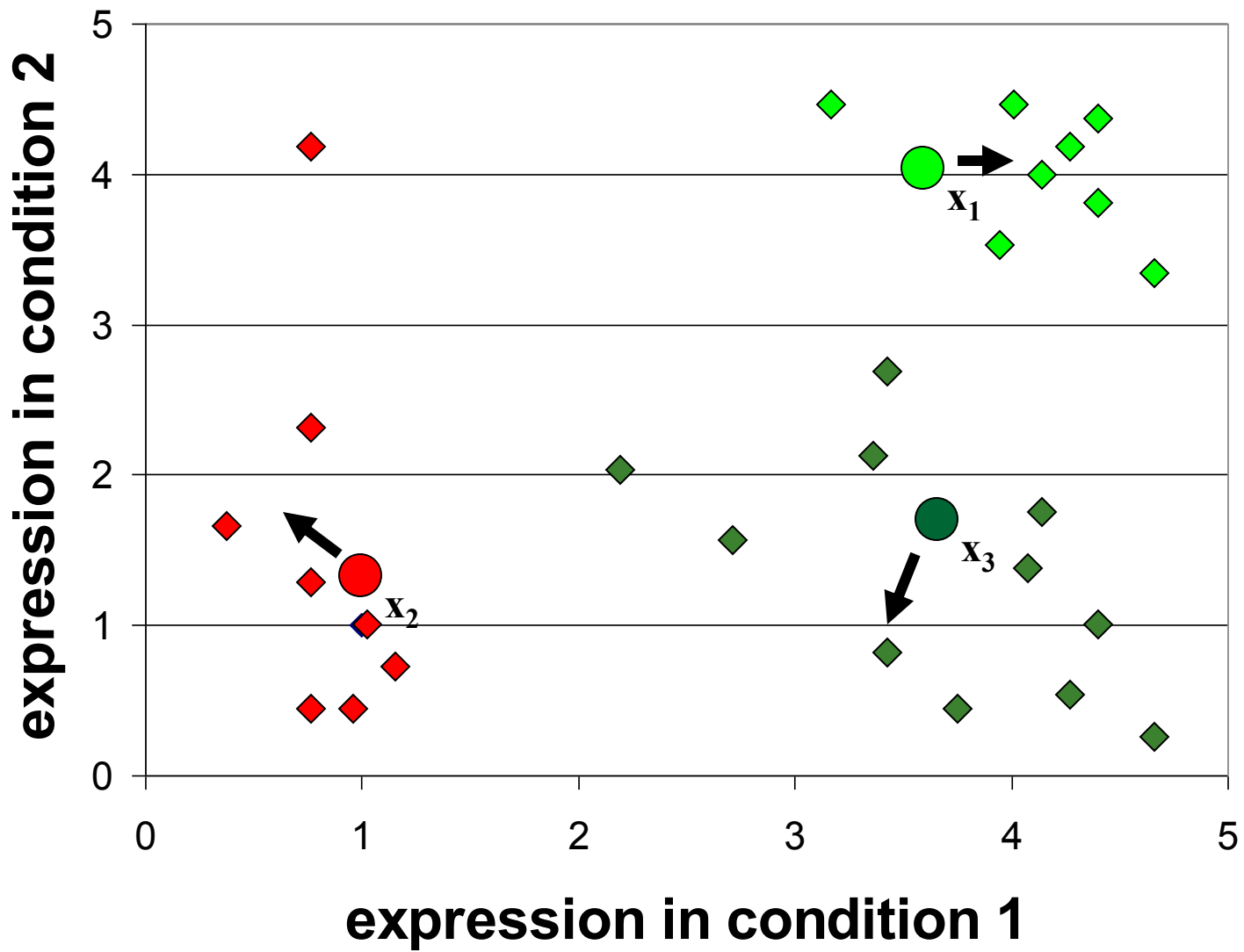
K-Means Clustering: Lloyd Algorithm

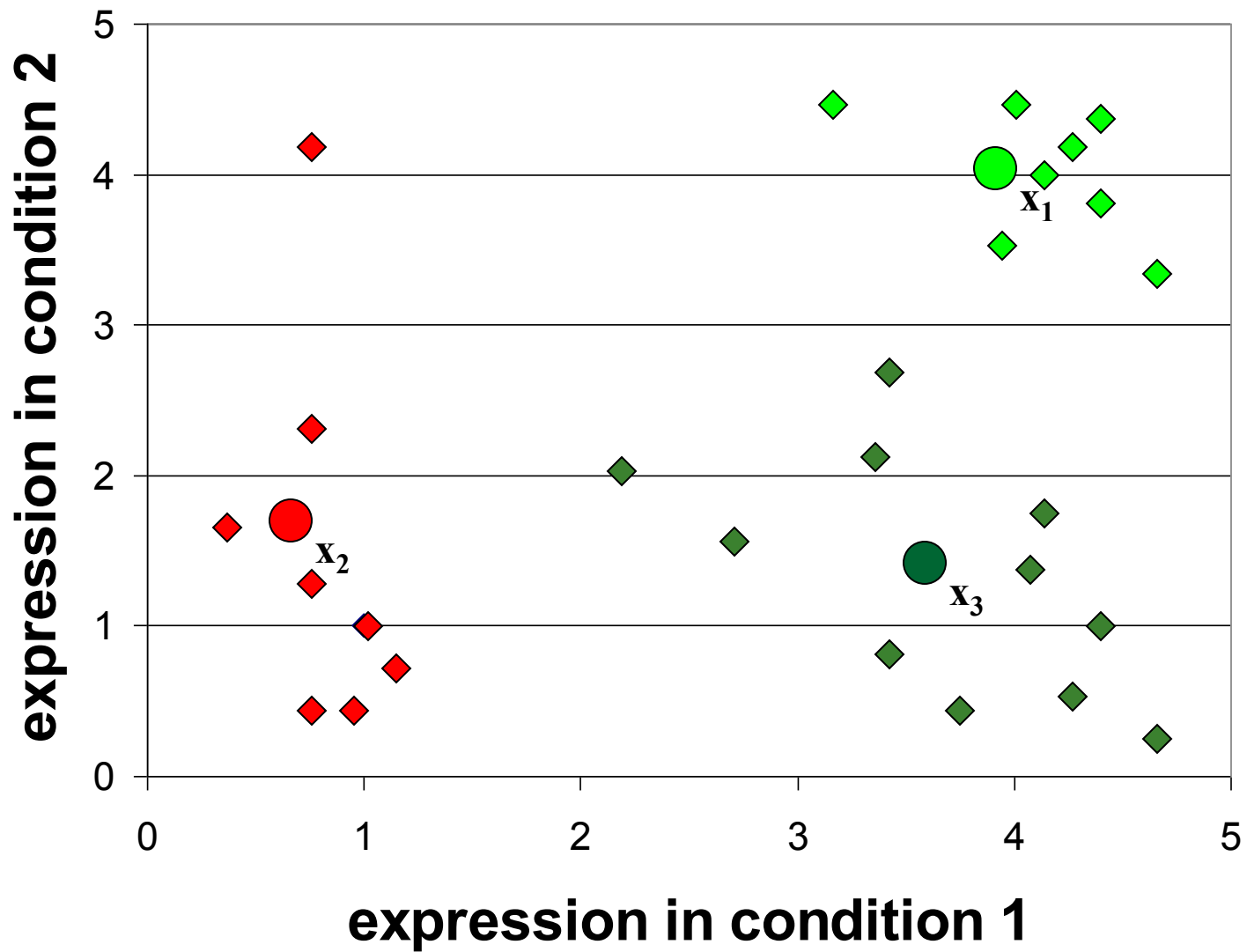
1. Lloyd Algorithm
2. Arbitrarily assign the k cluster centers
3. **while** the cluster centers keep changing
4. Assign each data point to the cluster C_i corresponding to the closest cluster representative (center) ($1 \leq i \leq k$)
5. After the assignment of all data points, compute new cluster representatives according to the center of gravity of each cluster, that is, the new cluster representative is
$$\frac{\sum v \in C}{|C|} \text{ for all } v \text{ in } C \text{ for every cluster } C$$

*This may lead to merely a locally optimal clustering.









Conservative K-Means Algorithm

- Lloyd algorithm is fast but in each iteration it moves many data points, not necessarily causing better convergence.
 - A more conservative method would be to move one point at a time only if it improves the overall **clustering cost**
 - The smaller the clustering cost of a partition of data points is the better that clustering is
 - Different methods (e.g., the squared error distortion) can be used to measure this clustering cost
-

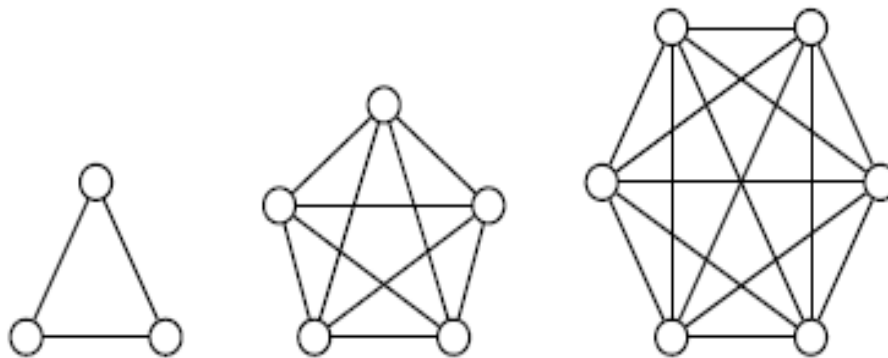
K-Means Greedy Algorithm

1. ProgressiveGreedyK-Means(k)
2. Select an arbitrary partition P into k clusters
3. **while** forever
4. $bestChange \leftarrow 0$
5. **for** every cluster C
6. **for** every element i not in C
7. **if** moving i to cluster C reduces its clustering cost
8. **if** $(\text{cost}(P) - \text{cost}(P_{i \rightarrow C}) > bestChange$
9. $bestChange \leftarrow \text{cost}(P) - \text{cost}(P_{i \rightarrow C})$
10. $i^* \leftarrow i$
11. $C^* \leftarrow C$
12. **if** $bestChange > 0$
13. Change partition P by moving i^* to C^*
14. **else**
15. **return** P

CLUSTERING USING GRAPHS

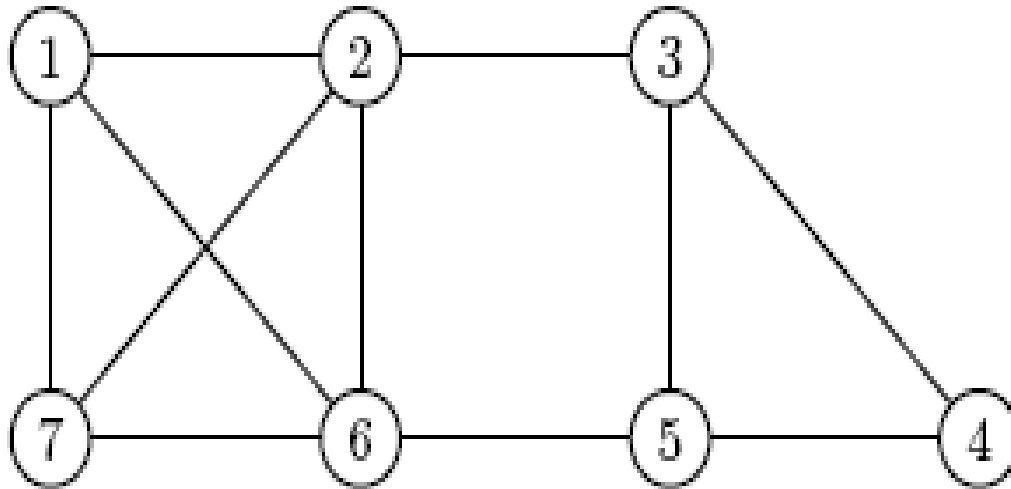
Clique Graphs

- A **clique** is a graph with every vertex connected to every other vertex
- A **clique graph** is a graph where each connected component is a clique



Transforming an Arbitrary Graph into a Clique Graphs

- A graph can be transformed into a clique graph by adding or removing edges



Corrupted Cliques Problem

Input: A graph G

Output: The smallest number of additions and removals of edges that will transform G into a clique graph

Distance Graphs

- Turn the distance matrix into a distance graph
 - Genes are represented as vertices in the graph
 - Choose a distance threshold θ
 - If the distance between two vertices is below θ , draw an edge between them
 - The resulting graph may contain cliques
 - These cliques represent clusters of closely located data points
-

Transforming Distance Graph into Clique Graph

The distance graph (threshold $\theta=7$) is transformed into a clique graph after removing the two highlighted edges

	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}
g_1	0.0	8.1	9.2	7.7	9.3	2.3	5.1	10.2	6.1	7.0
g_2	8.1	0.0	12.0	0.9	12.0	9.5	10.1	12.8	2.0	1.0
g_3	9.2	12.0	0.0	11.2	0.7	11.1	8.1	1.1	10.5	11.5
g_4	7.7	0.9	11.2	0.0	11.2	9.2	9.5	12.0	1.6	1.1
g_5	9.3	12.0	0.7	11.2	0.0	11.2	8.5	1.0	10.6	11.6
g_6	2.3	9.5	11.1	9.2	11.2	0.0	5.6	12.1	7.7	8.5
g_7	5.1	10.1	8.1	9.5	8.5	5.6	0.0	9.1	8.3	9.3
g_8	10.2	12.8	1.1	12.0	1.0	12.1	9.1	0.0	11.4	12.4
g_9	6.1	2.0	10.5	1.6	10.6	7.7	8.3	11.4	0.0	1.1
g_{10}	7.0	1.0	11.5	1.1	11.6	8.5	9.3	12.4	1.1	0.0

(a) Distance matrix, d (distances shorter than 7 are shown in bold).

After transforming the distance graph into the clique graph, the dataset is partitioned into three clusters

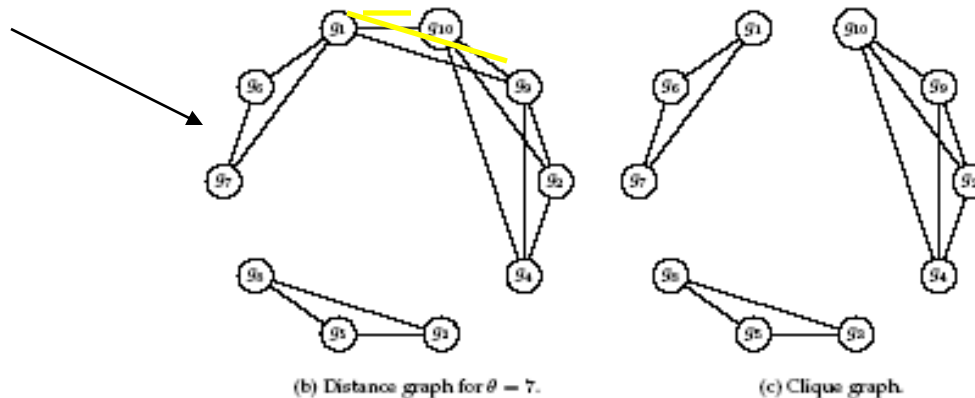


Figure 10.6 The distance graph (b) for $\theta = 7$ is not quite a clique graph. However, it can be transformed into a clique graph (c) by removing edges (g_1, g_{10}) and (g_1, g_9) .

Heuristics for Corrupted Clique Problem

- Corrupted Cliques problem is NP-Hard, some heuristics exist to approximately solve it:
- **CAST** (Cluster Affinity Search Technique): a practical and fast algorithm:
 - **CAST** is based on the notion of genes *close* to cluster C or *distant* from cluster C
 - Distance between gene i and cluster C :

$d(i, C)$ = average distance between gene i and all genes in C

Gene i is *close* to cluster C if $d(i, C) < \theta$ and *distant* otherwise

CAST Algorithm

1. CAST(S, G, θ)
2. $P \leftarrow \emptyset$
3. while $S \neq \emptyset$
4. $V \leftarrow$ vertex of maximal degree in the distance graph G
5. $C \leftarrow \{V\}$
6. while a **close** gene i **not in** C or **distant** gene i **in** C exists
7. Find the nearest close gene i not in C and add it to C
8. Remove the farthest distant gene i in C
9. Add cluster C to partition P
10. $S \leftarrow S \setminus C$
11. Remove vertices of cluster C from the distance graph G
12. return P

S – set of elements, G – distance graph, θ – distance threshold