Clustering algorithms

Konstantinos Koutroumbas

Unit 8

– Hierarchical clustering algorithms
  * Matrix theory-based
  * Graph theory-based

koutroum@noa.gr

Agglomerative Clustering Algorithms

According to the mathematical tools used for their expression, **agglomerative algorithms** are divided into:

- Algorithms based on matrix theory.
- Algorithms based on graph theory.

NOTE: In the sequel we consider only dissimilarity measures.

➢ Algorithms based on matrix theory.

- They take as input the $N \times N$ dissimilarity matrix $P_0 = P(X)$.

- At each level t where two clusters C_i and C_j are **merged** to C_q, the dissimilarity matrix P_t is extracted from P_{t-1} by:

  – **Deleting** the two rows and columns of P_t that **correspond** to C_i and C_j.

  – **Adding** a new row and a new column that contain the distances of newly formed $C_q = C_i \cup C_j$ from each of the remaining clusters C_s, via a relation of the form

$$d(C_q, C_s) = f(d(C_i, C_s), d(C_j, C_s), d(C_i, C_j))$$

Agglomerative matrix theory based Clustering Algorithms

- A number of distance functions comply with the following update equation

$C_q = C_i \cup C_j$

$$d(C_q, C_s) = a_i d(C_i, C_s) + a_j (d(C_j, C_s) + b d(C_i, C_j) + c|d(C_i, C_s) - d(C_j, C_s)| \quad \textbf{(1)}$$

Algorithms that follow the above equation are:

➢Single link (SL) algorithm ($a_i = 1/2, a_j = 1/2, b = 0, c = -1/2$). In this case

$$d(C_q, C_s) = \min\{d(C_i, C_s), d(C_j, C_s)\} \quad \textbf{(2)}$$

➢Complete link (CL) algorithm ($a_i = 1/2, a_j = 1/2, b = 0, c = 1/2$). In this case

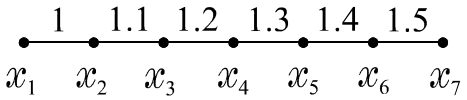
$$d(C_q, C_s) = \max\{d(C_i, C_s), d(C_j, C_s)\}$$

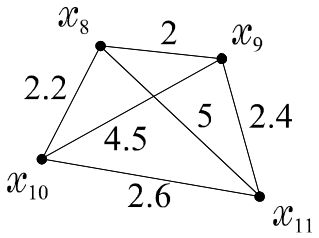
Remarks:

- Single link forms clusters at low dissimilarities while complete link forms clusters at high dissimilarities.
- Single link tends to form elongated clusters (*chaining effect*) while complete link tends to form compact clusters.
- The rest algorithms are compromises between these two extremes.

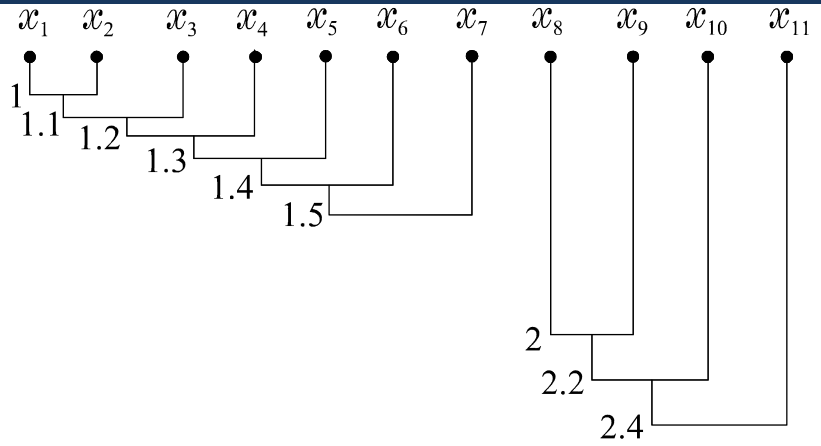
Agglomerative matrix theory based Clustering Algorithms

Example:





(a)

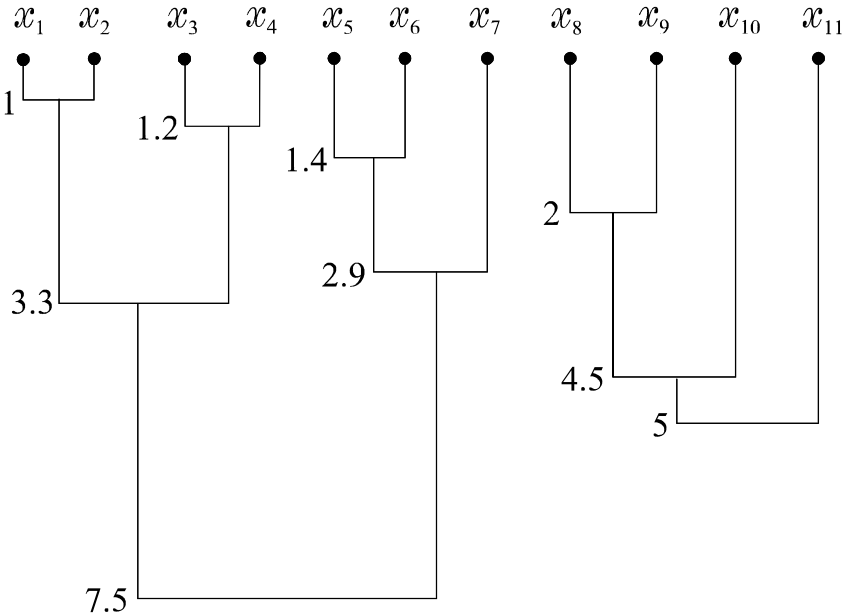


(b)

(a) The data set X.

(b) The single link algorithm dissimilarity dendrogram.

(c) The complete link algorithm dissimilarity dendrogram.



(c)

Agglomerative matrix theory based Clustering Algorithms

- Weighted Pair Group Method Average (WPGMA) ($a_i = 1/2, a_j = 1/2, b = 0, c = 0$). In this case:

$$d(C_q, C_s) = a_i d(C_i, C_s) + a_j (d(C_j, C_s) + b d(C_i, C_j) + c|d(C_i, C_s) - d(C_j, C_s)|$$

$$d(C_q, C_s) = \frac{1}{2}(d(C_i, C_s) + d(C_j, C_s))$$

- Unweighted Pair Group Method Average (UPGMA) ($a_i = n_i/(n_i + n_j), a_j = n_j/(n_i + n_j), b = 0, c = 0$, where n_i is the cardinality of C_i). In this case:

$$d(C_q, C_s) = \frac{n_i}{n_i + n_j} d(C_i, C_s) + \frac{n_j}{n_i + n_j} d(C_j, C_s)$$

- Unweighted Pair Group Method Centroid (UPGMC) ($a_i = n_i/(n_i + n_j)$, $a_j = n_j/(n_i + n_j), b = -n_i n_j/(n_i + n_j)^2, c = 0$). In this case:

$$d_{qs} = \frac{n_i}{n_i + n_j} d_{is} + \frac{n_j}{n_i + n_j} d_{js} - \frac{n_i n_j}{(n_i + n_j)^2} d_{ij}$$

For the UPGMC, **if** d_{ij} is defined as the squared Euclidean distance between the means of C_i and C_j,
then it holds that $d_{qs} = ||\boldsymbol{m}_q - \boldsymbol{m}_s||^2$, where $\boldsymbol{m}_q$, $\boldsymbol{m}_s$ are the means of C_q, C_s, respectively.

Agglomerative matrix theory based Clustering Algorithms

- Weighted Pair Group Method Centroid (WPGMC) ($a_i = 1/2, a_j = 1/2, b = -1/4, c = 0$). In this case

$$d_{qs} = \frac{1}{2} d_{is} + \frac{1}{2} d_{js} - \frac{1}{4} d_{ij}$$

$$d(C_q, C_s) = a_i d(C_i, C_s) + a_j (d(C_j, C_s) + bd(C_i, C_j) + c|d(C_i, C_s) - d(C_j, C_s)|$$

For WPGMC there are cases where $d_{qs} \leq \max\{d_{is}, d_{js}\}$ (crossover)

- Ward or minimum variance algorithm. Here the distance d'_{ij} between C_i and C_j is defined as

$$d'_{ij} = \frac{n_i n_j}{n_i + n_j} ||\boldsymbol{m}_i - \boldsymbol{m}_j||^2 \qquad \textbf{(3)}$$

d'_{qs} can be expressed in terms of $d'_{is}, d'_{js}, d'_{ij}$ as

$$d'_{qs} = \frac{n_i + n_s}{n_i + n_j + n_s} d'_{is} + \frac{n_j + n_s}{n_i + n_j + n_s} d'_{js} - \frac{n_s}{n_i + n_j + n_s} d'_{ij}$$

Remark: Ward's algorithm forms $\Re_{t+1}$ by merging the two clusters that lead to the smallest possible increase of the total variance, i.e.,

$$E_t = \sum_{r=1}^{N-t} \sum_{\boldsymbol{x} \in C_r} ||\boldsymbol{x} - \boldsymbol{m}_r||^2$$

Agglomerative matrix theory based Clustering Algorithms

Example 3: Consider the following dissimilarity matrix (Euclidean distance)

$$P_0 = \begin{bmatrix} 0 & 1 & 2 & 26 & 37 \\ 1 & 0 & 3 & 25 & 36 \\ 2 & 3 & 0 & 16 & 25 \\ 26 & 25 & 16 & 0 & 1.5 \\ 37 & 36 & 25 & 1.5 & 0 \end{bmatrix}$$

$\mathcal{R}_0 = \{\{\underline{x}_1\}, \{\underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4\}, \{\underline{x}_5\}\}$,
$\mathcal{R}_1 = \{\{\underline{x}_1, \underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4\}, \{\underline{x}_5\}\}$,
$\mathcal{R}_2 = \{\{\underline{x}_1, \underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4, \underline{x}_5\}\}$,
$\mathcal{R}_3 = \{\{\underline{x}_1, \underline{x}_2, \underline{x}_3\}, \{\underline{x}_4, \underline{x}_5\}\}$,
$\mathcal{R}_4 = \{\{\underline{x}_1, \underline{x}_2, \underline{x}_3, \underline{x}_4, \underline{x}_5\}\}$

All the algorithms produce the same sequence of clusterings shown above, yet at different proximity levels:

	SL	*CL*	*WPGMA*	*UPGMA*	*WPGMC*	*UPGMC*	*Ward*
$\mathcal{R}_0$	0	0	0	0	0	0	0
$\mathcal{R}_1$	1	1	1	1	1	1	0.5
$\mathcal{R}_2$	1.5	1.5	1.5	1.5	1.5	1.5	0.75
$\mathcal{R}_3$	2	3	2.5	2.5	2.25	2.25	1.5
$\mathcal{R}_4$	16	37	25.75	27.5	24.69	26.46	31.75

Agglomerative matrix theory based Clustering Algorithms

Example 3 (in detail): (a) The single-link case

$(C_q = C_i \cup C_j,\ d(C_q, C_s) = \min(d(C_i, C_s), d(C_j, C_s))$

P_0:

	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$
$\{x_1\}$	0	**1**	2	26	37
$\{x_2\}$	**1**	0	3	25	36
$\{x_3\}$	2	3	0	16	25
$\{x_4\}$	26	25	16	0	1.5
$\{x_5\}$	37	36	25	1.5	0

	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$
$\{x_1\}$	0	**1**	2	26	37
$\{x_2\}$	**1**	0	3	25	36
$\{x_3\}$	2	3	0	16	25
$\{x_4\}$	26	25	16	0	1.5
$\{x_5\}$	37	36	25	1.5	0

$d(\{x_1, x_2\}, \{x_3\}) = \min(d(\{x_1\}, \{x_3\}), d(\{x_2\}, \{x_3\})) = \min(2,3) = 2$

$d(\{x_1, x_2\}, \{x_4\}) = \min(26,25) = 25$

$d(\{x_1, x_2\}, \{x_5\}) = \min(37,36) = 36$

P_1:

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$
$\{x_1, x_2\}$	0	2	25	36
$\{x_3\}$	2	0	16	25
$\{x_4\}$	25	16	0	**1.5**
$\{x_5\}$	36	25	**1.5**	0

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$
$\{x_1, x_2\}$	0	2	25	36
$\{x_3\}$	2	0	16	25
$\{x_4\}$	25	16	0	**1.5**
$\{x_5\}$	36	25	**1.5**	0

$d(\{x_1, x_2\}, \{x_4, x_5\}) = \min(25,36) = 25$

$d(\{x_3\}, \{x_4, x_5\}) = \min(16,25) = 16$

Agglomerative matrix theory based Clustering Algorithms

Example 3 (in detail): (a) The single-link case

$(C_q = C_i \cup C_j,\ d(C_q, C_s) = \min(d(C_i, C_s), d(C_j, C_s))$

P_2:

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4, x_5\}$
$\{x_1, x_2\}$	0	**2**	25
$\{x_3\}$	**2**	0	16
$\{x_4, x_5\}$	25	16	0

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4, x_5\}$
$\{x_1, x_2\}$	0	**2**	25
$\{x_3\}$	**2**	0	16
$\{x_4, x_5\}$	25	16	0

$d(\{x_1, x_2, x_3\}, \{x_4, x_5\}) = \min(25,16) = 16$

P_3:

	$\{x_1, x_2, x_3\}$	$\{x_4, x_5\}$
$\{x_1, x_2, x_3\}$	0	**16**
$\{x_4, x_5\}$	**16**	0

	$\{x_1, x_2, x_3\}$	$\{x_4, x_5\}$
$\{x_1, x_2, x_3\}$	0	**16**
$\{x_4, x_5\}$	**16**	0

P_4:

	$\{x_1, x_2, x_3, x_4, x_5\}$
$\{x_1, x_2, x_3, x_4, x_5\}$	0

$\mathcal{R}_0 = \{\{\underline{x}_1\}, \{\underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4\}, \{\underline{x}_5\}\}, (\mathbf{0})$

$\mathcal{R}_1 = \{\{\underline{x}_1, \underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4\}, \{\underline{x}_5\}\}, (\mathbf{1})$

$\mathcal{R}_2 = \{\{\underline{x}_1, \underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4, \underline{x}_5\}\}, (\mathbf{1.5})$

$\mathcal{R}_3 = \{\{\underline{x}_1, \underline{x}_2, \underline{x}_3\}, \{\underline{x}_4, \underline{x}_5\}\}, (\mathbf{2})$

$\mathcal{R}_4 = \{\{\underline{x}_1, \underline{x}_2, \underline{x}_3, \underline{x}_4, \underline{x}_5\}\}, (\mathbf{16})$

Agglomerative matrix theory based Clustering Algorithms

Example 3 (in detail): (b) The complete-link case

$(C_q = C_i \cup C_j,\ d(C_q, C_s) = \max(d(C_i, C_s), d(C_j, C_s))$

P_0:

	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$
$\{x_1\}$	0	**1**	2	26	37
$\{x_2\}$	**1**	0	3	25	36
$\{x_3\}$	2	3	0	16	25
$\{x_4\}$	26	25	16	0	1.5
$\{x_5\}$	37	36	25	1.5	0

	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$
$\{x_1\}$	0	**1**	2	26	37
$\{x_2\}$	**1**	0	3	25	36
$\{x_3\}$	2	3	0	16	25
$\{x_4\}$	26	25	16	0	1.5
$\{x_5\}$	37	36	25	1.5	0

$d(\{x_1, x_2\}, \{x_3\}) = \max(d(\{x_1\}, \{x_3\}), d(\{x_2\}, \{x_3\})) = \max(2,3) = 3$

$d(\{x_1, x_2\}, \{x_4\}) = \max(26,25) = 26$

$d(\{x_1, x_2\}, \{x_5\}) = \max(37,36) = 37$

P_1:

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$
$\{x_1, x_2\}$	0	3	26	37
$\{x_3\}$	3	0	16	25
$\{x_4\}$	26	16	0	**1.5**
$\{x_5\}$	37	25	**1.5**	0

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$
$\{x_1, x_2\}$	0	3	26	37
$\{x_3\}$	3	0	16	25
$\{x_4\}$	26	16	0	**1.5**
$\{x_5\}$	37	25	**1.5**	0

$d(\{x_1, x_2\}, \{x_4, x_5\}) = \max(26,37) = 37$

$d(\{x_3\}, \{x_4, x_5\}) = \max(16,25) = 25$

Agglomerative matrix theory based Clustering Algorithms

Example 3 (in detail): (b) The complete-link case

$(C_q = C_i \cup C_j,\ d(C_q, C_s) = \max(d(C_i, C_s), d(C_j, C_s))$

P_2:

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4, x_5\}$
$\{x_1, x_2\}$	0	**3**	37
$\{x_3\}$	**3**	0	25
$\{x_4, x_5\}$	37	25	0

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4, x_5\}$
$\{x_1, x_2\}$	0	**3**	37
$\{x_3\}$	**3**	0	25
$\{x_4, x_5\}$	37	25	0

$d(\{x_1, x_2, x_3\}, \{x_4, x_5\}) =$
$= \max(37,25) = 37$

P_3:

	$\{x_1, x_2, x_3\}$	$\{x_4, x_5\}$
$\{x_1, x_2, x_3\}$	0	**37**
$\{x_4, x_5\}$	**37**	0

	$\{x_1, x_2, x_3\}$	$\{x_4, x_5\}$
$\{x_1, x_2, x_3\}$	0	**37**
$\{x_4, x_5\}$	**37**	0

P_4:

	$\{x_1, x_2, x_3, x_4, x_5\}$
$\{x_1, x_2, x_3, x_4, x_5\}$	0

$\mathcal{R}_0 = \{\{\underline{x}_1\}, \{\underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4\}, \{\underline{x}_5\}\}, (\mathbf{0})$

$\mathcal{R}_1 = \{\{\underline{x}_1, \underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4\}, \{\underline{x}_5\}\}, (\mathbf{1})$

$\mathcal{R}_2 = \{\{\underline{x}_1, \underline{x}_2\}, \{\underline{x}_3\}, \{\underline{x}_4, \underline{x}_5\}\}, (\mathbf{1.5})$

$\mathcal{R}_3 = \{\{\underline{x}_1, \underline{x}_2, \underline{x}_3\}, \{\underline{x}_4, \underline{x}_5\}\}, (\mathbf{3})$

$\mathcal{R}_4 = \{\{\underline{x}_1, \underline{x}_2, \underline{x}_3, \underline{x}_4, \underline{x}_5\}\}, (\mathbf{37})$

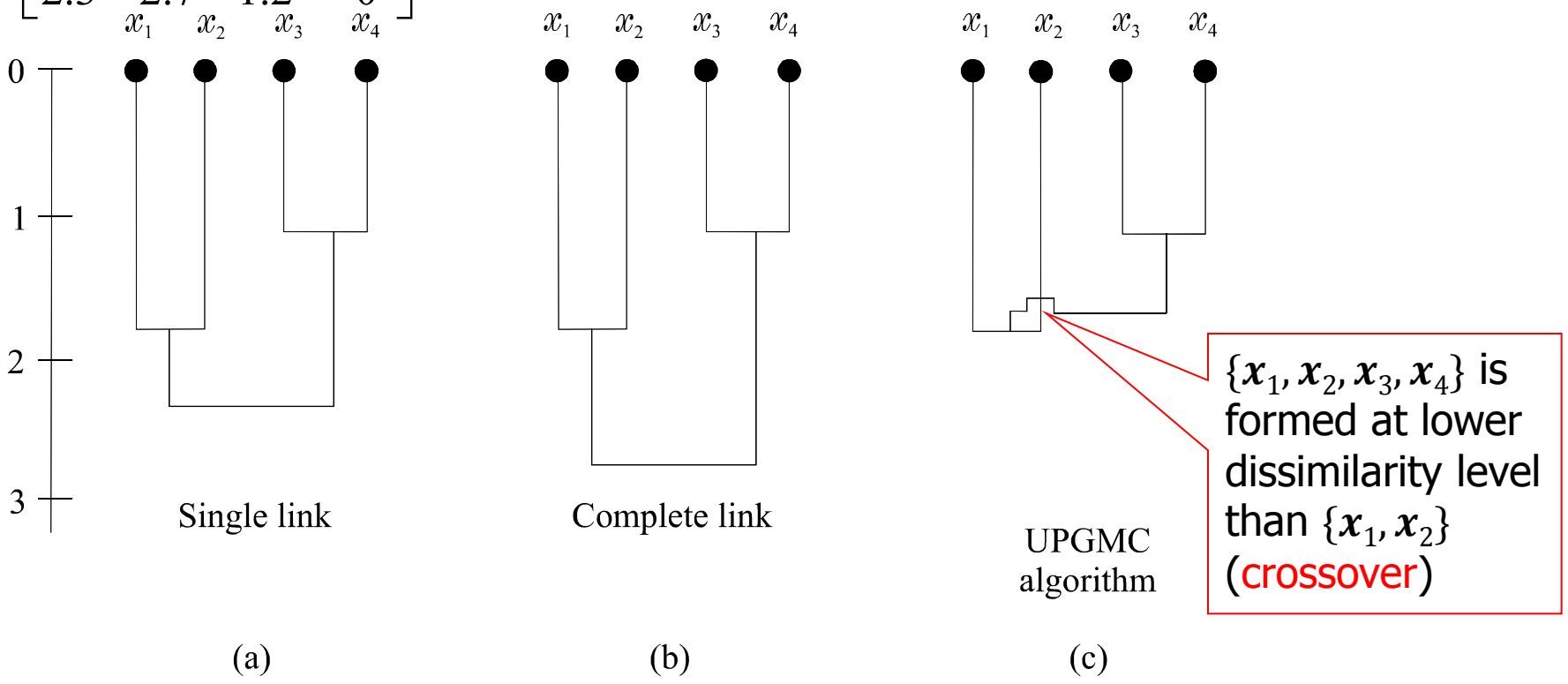
Agglomerative matrix theory based Clustering Algorithms

Monotonicity and crossover:

For the following dissimilarity matrix

$$P = \begin{bmatrix} 0 & 1.8 & 2.4 & 2.3 \\ 1.8 & 0 & 2.5 & 2.7 \\ 2.4 & 2.5 & 0 & 1.2 \\ 2.3 & 2.7 & 1.2 & 0 \end{bmatrix}$$

the dissimilarity dendrograms produced by single link, complete link and UPGMC (the same result is produced if WPGMC is employed) are:



Agglomerative matrix theory based Clustering Algorithms

Example (in detail): The WPGMC case

$(C_q = C_i \cup C_j,\ d_{qs} = \frac{1}{2}d_{is} + \frac{1}{2}d_{js} - \frac{1}{4}d_{ij})$

P_0:

	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_4\}$
$\{x_1\}$	0	1.8	2.4	2.3
$\{x_2\}$	1.8	0	2.5	2.7
$\{x_3\}$	2.4	2.5	0	**1.2**
$\{x_4\}$	2.3	2.7	**1.2**	0

	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_4\}$
$\{x_1\}$	0	1.8	2.4	2.3
$\{x_2\}$	1.8	0	2.5	2.7
$\{x_3\}$	2.4	2.5	0	**1.2**
$\{x_4\}$	2.3	2.7	**1.2**	0

$$d_{(3,4),1} = \frac{1}{2}d_{3,1} + \frac{1}{2}d_{4,1} - \frac{1}{4}d_{3,4} = \frac{1}{2}2.4 + \frac{1}{2}2.3 - \frac{1}{4}1.2 = 2.05$$

$$d_{(3,4),2} = \frac{1}{2}d_{3,2} + \frac{1}{2}d_{4,2} - \frac{1}{4}d_{3,4} = \frac{1}{2}2.5 + \frac{1}{2}2.7 - \frac{1}{4}1.2 = 2.3$$

P_1:

	$\{x_1\}$	$\{x_2\}$	$\{x_3, x_4\}$
$\{x_1\}$	0	**1.8**	2.05
$\{x_2\}$	**1.8**	0	2.3
$\{x_3, x_4\}$	2.05	2.3	0

	$\{x_1\}$	$\{x_2\}$	$\{x_3, x_4\}$
$\{x_1\}$	0	**1.8**	2.05
$\{x_2\}$	**1.8**	0	2.3
$\{x_3, x_4\}$	2.05	2.3	0

$\mathcal{R}_0 = \{\{\underline{x}_1\},\{\underline{x}_2\},\{\underline{x}_3\},\{\underline{x}_4\}\}$, **(0)**

$\mathcal{R}_1 = \{\{\underline{x}_1\},\{\underline{x}_2\},\{\underline{x}_3,\underline{x}_4\}\}$, **(1.2)**

$\mathcal{R}_2 = \{\{\underline{x}_1,\underline{x}_2\},\{\underline{x}_3,\underline{x}_4\}\}$, **(1.8)**

$\mathcal{R}_3 = \{\{\underline{x}_1,\underline{x}_2,\underline{x}_3,\underline{x}_4\}\}$, **(1.275 !!)**

$$d_{(1,2),(3,4)} = \frac{1}{2}d_{1,(3,4)} + \frac{1}{2}d_{2,(3,4)} - \frac{1}{4}d_{1,2} = \frac{1}{2}2.05 + \frac{1}{2}2.3 - \frac{1}{4}1.8 = 1.275$$

P_2:

	$\{x_1, x_2\}$	$\{x_3, x_4\}$
$\{x_1, x_2\}$	0	**1.275**
$\{x_3, x_4\}$	**1.275**	0

P_3:

	$\{x_1, x_2, x_3, x_4\}$
$\{x_1, x_2, x_3, x_4\}$	0

Agglomerative matrix theory based Clustering Algorithms

➢ Monotonicity condition:

If clusters C_i and C_j are selected to be merged in cluster C_q, at the tth level of the hierarchy, the condition

$$d(C_q, C_k) \geq d(C_i, C_j)$$

must hold for all $C_k, k \neq i, j, q$.

In other words, the monotonicity condition implies that a clustering is formed at higher dissimilarity level than any of its components.

Remarks:

- Monotonicity is a property that is **exclusively related** to the clustering algorithm and not to the (initial) proximity matrix.
- An algorithm that does not **satisfy** the monotonicity condition, does not necessarily **produce** dendrograms with crossovers.
- Single link, complete link, UPGMA, WPGMA and the Ward's algorithm **satisfy** the monotonicity condition, while UPGMC and WPGMC **do not satisfy** it.

Agglomerative matrix theory based Clustering Algorithms

Complexity issues:

- GAS requires, in general, $O(N^3)$ operations.
- More efficient implementations require $O(N^2 \log N)$ computational time.
- For a class of widely used algorithms, implementations that require $O(N^2)$ computational time and $O(N^2)$ or $O(N)$ storage have also been proposed.
- Parallel implementations on SIMD machines have also been considered.

Agglomerative graph theory based Clustering Algorithms

Some basic definitions from graph theory:

- A graph, G, is defined as an ordered pair $G = (V, E)$, where $V = \{v_i, i = 1, \dots, N\}$ is a set of vertices and E is a set of edges connecting some pairs of vertices. An edge connecting v_i and v_j is denoted by e_{ij} or (v_i, v_j).

- A graph is called undirected if there is no direction assigned to any of its edges. Otherwise, we deal with directed graphs.

- A graph is called unweighted if there is no cost associated with any of its edges. Otherwise, we deal with weighted graphs.

- A path in G between vertices v_{i_1} and v_{i_n} is a sequence of **vertices** and **edges** of the form $v_{i_1} e_{i_1 i_2} v_{i_2} \dots v_{i_{n-1}} e_{i_{n-1} i_n} v_{i_n}$.

- A loop in G is a path where v_{i_1} and v_{i_n} coincide.

- A subgraph $G' = (V', E')$ of $G = (V, E)$ is a graph with $V' \subseteq V$ and $E' \subseteq E_1$, where E_1 is a subset of E containing edges that connect vertices of V'. Every graph is a subgraph to itself.

Agglomerative graph theory based Clustering Algorithms

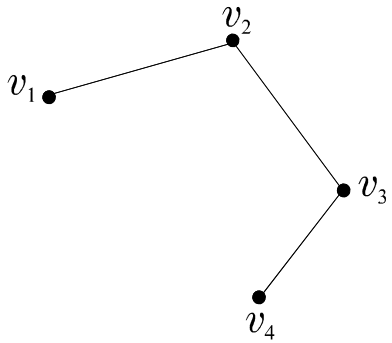
Some basic definitions from graph theory (cont.):

- A connected subgraph $G' = (V', E')$ is a subgraph where there exists at least one path connecting any pair of vertices in V'.
- A complete subgraph $G' = (V', E')$ is a subgraph where for any pair of vertices in V' there exists an edge in E' connecting them.
- A maximally connected subgraph of G is a connected subgraph G' of G that contains as many vertices of G as possible.
- A maximally complete subgraph of G is a complete subgraph G' of G that contains as many vertices of G as possible.

Examples for the above, are shown in the following figure.

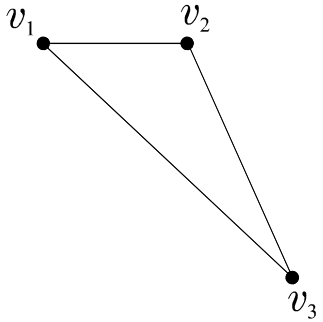
Agglomerative graph theory based Clustering Algorithms

Some basic definitions from graph theory (cont.):



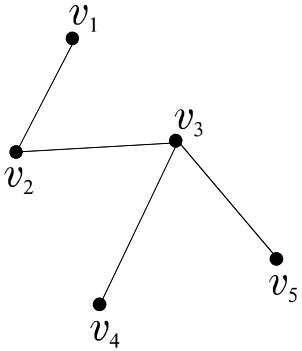
Path

(a)



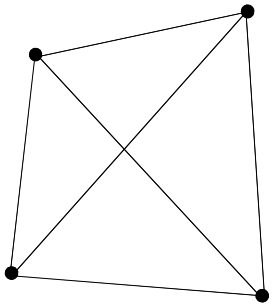
Loop

(b)



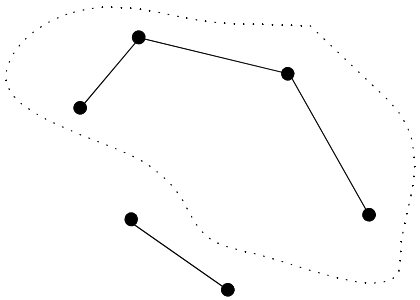
Connected graph

(c)



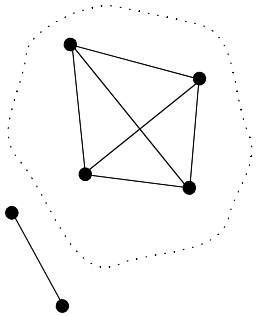
Complete graph

(d)



Maximally connected subgraph

(e)



Maximally complete subgraph

(f)

Agglomerative graph theory based Clustering Algorithms

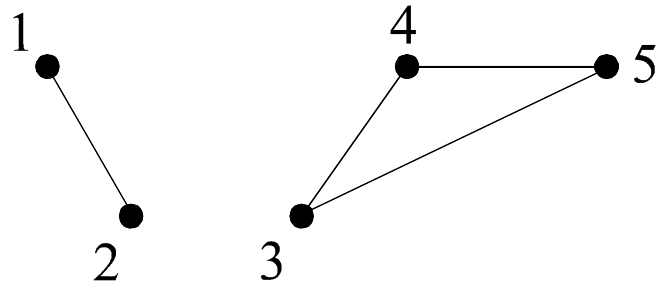
NOTE: In the framework of clustering, each vertex of a graph corresponds to a feature vector.

Useful **tools** for the algorithms based on graph theory are the threshold graph and the proximity graph.

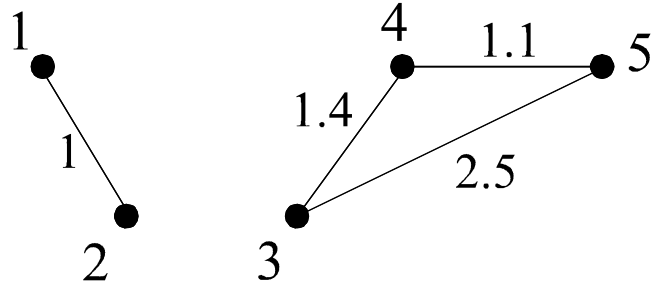
- A threshold graph $G(a)$ (a is the threshold parameter)
  - is an undirected, unweighted graph with N nodes, each one corresponding to a vector of X.
  - No self-loops or multiple edges between any two vertices are encountered.
  - The set of edges of $G(a)$ contains those edges (v_i, v_j) for which the distance $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ between the vectors corresponding to v_i and v_j is less than or equal to a.

- A proximity graph $G_p(a)$ is a threshold graph $G(a)$, all of whose edges (v_i, v_j) are weighted with the proximity measure $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Agglomerative graph theory based Clustering Algorithms

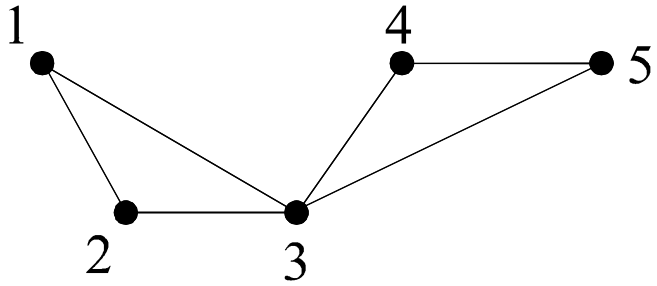


(a)

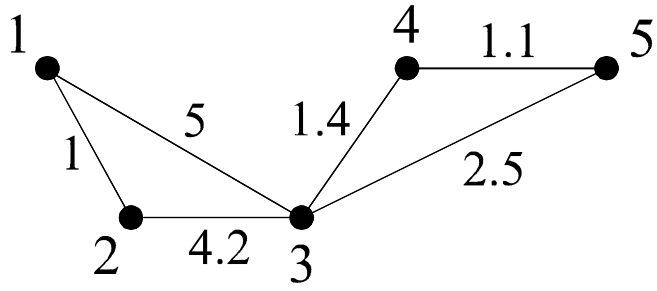


(b)

$$P(X) = \begin{bmatrix} 0 & 1 & 5 & 6.4 & 7.4 \\ 1 & 0 & 4.2 & 5.7 & 6.7 \\ 5 & 4.2 & 0 & 1.4 & 2.5 \\ 6.4 & 5.7 & 1.4 & 0 & 1.1 \\ 7.4 & 6.7 & 2.5 & 1.1 & 0 \end{bmatrix}$$



(c)



(d)

(a) The threshold graph $G(3)$, (b) the proximity (dissimilarity) graph $G_p(3)$, (c) the threshold graph $G(5)$, (d) the dissimilarity graph $G_p(5)$, for the dissimilarity matrix $P(X)$ shown above.

Agglomerative graph theory based Clustering Algorithms

More definitions:

- In this framework, we consider graphs G, of N nodes, where each node corresponds to a vector of X.
- Valid clusters are connected components of G that satisfy an **additional** graph property $h(k)$.

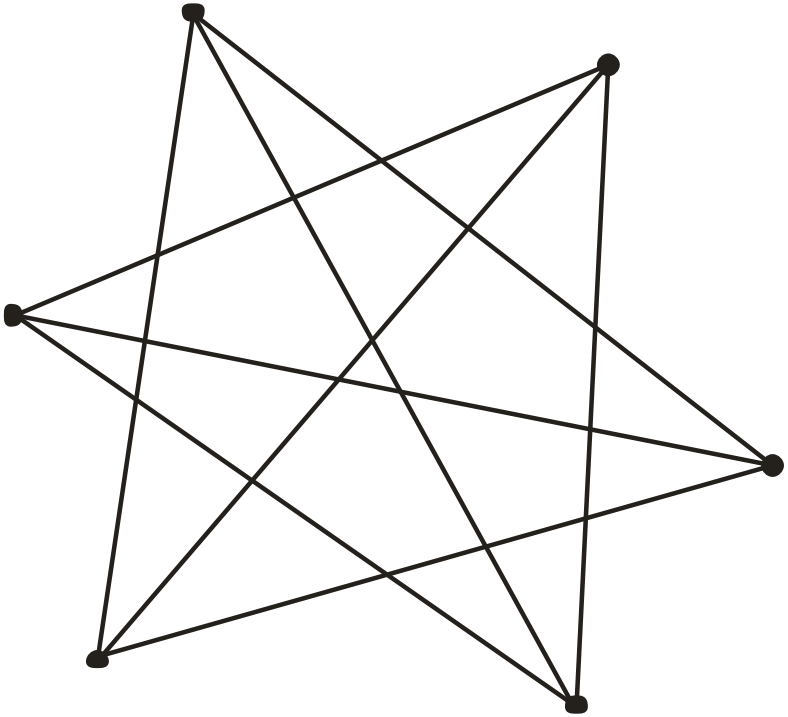
Typical graph properties for a connected component (subgraph) G' of G are:

- Node connectivity: The largest integer k such that all pairs of nodes of G' are joined by at least k paths having no nodes in common.

- Edge connectivity: The largest integer k such that all pairs of nodes are joined by at least k paths having no edges in common.

- Node degree: The largest integer k such that each node has at least k incident edges.

Agglomerative graph theory based Clustering Algorithms



Node connectivity : 3

Edge connectivity : 3

Node degree : 3

Agglomerative graph theory based Clustering Algorithms

➢ Proximity function in the graph theory framework

- The **proximity function** $g_{h(k)}(C_r, C_s)$ between two clusters is **defined** in terms of

– a proximity measure between vectors (nodes)

– certain constraints **imposed** by property $h(k)$ on the subgraphs that are formed.

In mathematical language:

$$g_{h(k)}(C_r, C_s) = \min_{\boldsymbol{x}_u \in C_r, \boldsymbol{x}_v \in C_s} \left\{ \begin{array}{c} d(\boldsymbol{x}_u, \boldsymbol{x}_v) \equiv a\text{: the } G(a) \text{ subgraph defined by } C_r \cup C_s \text{ is} \\ (a) \textbf{ connected and either } (b1) \text{ has the } \textbf{property } h(k) \text{ or } (b2) \text{ is } \textbf{complete} \end{array} \right\} \quad \textbf{(4)}$$

or

$g_{h(k)}(C_r, C_s)$ equals to the **smallest** possible value a such that in *the* $G(a)$ *subgraph defined by* $C_r \cup C_s$ *is* (a) ***connected*** *and either* $(b1)$ *has the* ***property*** $h(k)$ *or* $(b2)$ *is* ***complete***.

Agglomerative graph theory based Clustering Algorithms

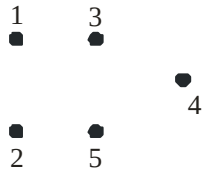
➢Example: For the dissimilarity matrix,

$$P = \begin{bmatrix} 0 & 1.2 & 3 & 3.7 & 4.2 \\ 1.2 & 0 & 2.5 & 3.2 & 3.9 \\ 3 & 2.5 & 0 & 1.8 & 2.0 \\ 3.7 & 3.2 & 1.8 & 0 & 1.5 \\ 4.2 & 3.9 & 2.0 & 1.5 & 0 \end{bmatrix}$$

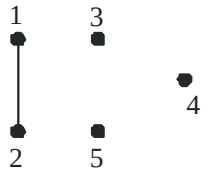
all possible $G(a)$ graphs are shown next.

Assuming that $h(2)$ is the node connectivity property, it is

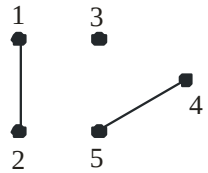
$g_h(\{x_1\},\{x_2\}) = 1.2$ (complete)
$g_h(\{x_1\},\{x_5\}) = 4.2$ (complete)
$g_h(\{x_1,x_2\},\{x_3\}) = 3$ (compl.-$h(2)$)
$g_h(\{x_1,x_2\},\{x_3,x_5\}) = 3.9$ ($h(2)$)



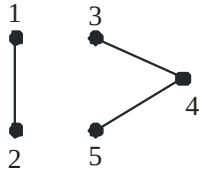
G(0)



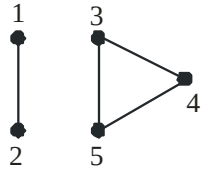
G(1.2)



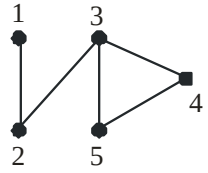
G(1.5)



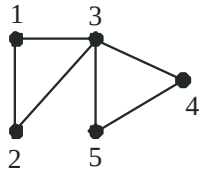
G(1.8)



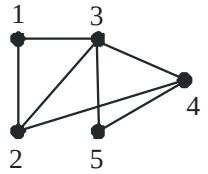
G(2.0)



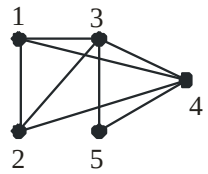
G(2.5)



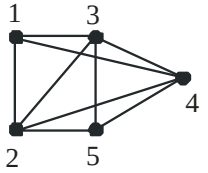
G(3.0)



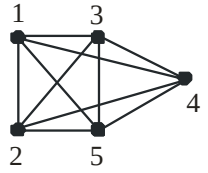
G(3.2)



G(3.7)



G(3.9)



G(4.2)

Agglomerative graph theory based Clustering Algorithms

➢Graph theory-based algorithmic scheme (GTAS): It is the GAS in the context of graph theory. In the context of GTAS, the definition of the proximity between the clusters is based on graph theory concepts. Thus

Generalized Agglomerative Scheme (GAS)

- Initialization
  - **Choose** $\Re_0 = \{\{\boldsymbol{x}_1\}, \ldots, \{\boldsymbol{x}_N\}\}$
  - $t = 0$
- Repeat
  - $t = t + 1$
  - **Choose** (C_i, C_j) in $\Re_{t-1}$ such that

$$g_{h(k)}(C_i, C_j) = \begin{cases} min_{r,s}\, g_{h(k)}(C_r, C_s), & for\ disim.\ functions \\ max_{r,s}\, g_{h(k)}(C_r, C_s), & for\ sim.\ functions \end{cases}$$

  - Define $C_q = C_i \cup C_j$ and produce $\Re_t = \left(\Re_{t-1} - \{C_i, C_j\}\right) \cup \{C_q\}$

- Until all vectors lie in a single cluster.

Agglomerative graph theory based Clustering Algorithms

- Single link (SL) algorithm. Here

$g_{h(k)}(C_r, C_s)$
$= min_{\boldsymbol{x}_u \in C_r, \boldsymbol{x}_v \in C_s}\{d(\boldsymbol{x}_u, \boldsymbol{x}_v) \equiv a\text{: the } G(a) \text{ subgraph defined by } C_r \cup C_s \text{ is } \boldsymbol{connected}\}$
$\equiv min_{\boldsymbol{x} \in C_r, \boldsymbol{y} \in C_s} d(\boldsymbol{x}, \boldsymbol{y})$ (why?)

- Remarks:
  - No property $h(k)$ or completeness is required.
  - The SL stemming from the graph theory is exactly the same with the SL stemming from the matrix theory.

- Complete link (CL) algorithm. Here

$g_{h(k)}(C_r, C_s)$
$= min_{\boldsymbol{x}_u \in C_r, \boldsymbol{x}_v \in C_s}\{d(\boldsymbol{x}_u, \boldsymbol{x}_v) \equiv a\text{: the } G(a) \text{ subgraph defined by } C_r \cup C_s \text{ is } \boldsymbol{complete}\}$
$\equiv max_{\boldsymbol{x} \in C_r, \boldsymbol{y} \in C_s} d(\boldsymbol{x}, \boldsymbol{y})$ (why?)

- Remarks:
  - No property $h(k)$ is required.
  - The CL stemming from graph theory is exactly the same with the CL stemming from matrix theory.

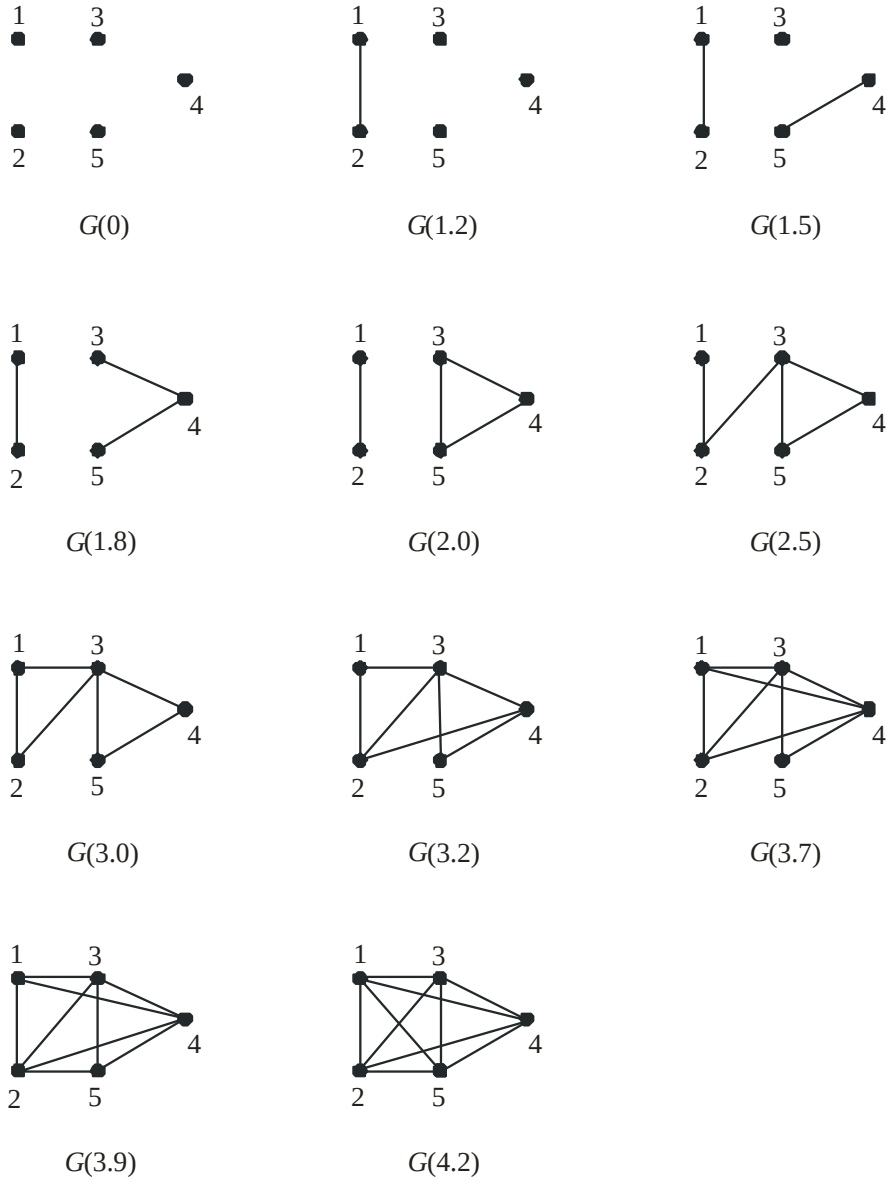
Agglomerative graph theory based Clustering Algorithms

➢Example: For the dissimilarity matrix,

$$P = \begin{bmatrix} 0 & 1.2 & 3 & 3.7 & 4.2 \\ 1.2 & 0 & 2.5 & 3.2 & 3.9 \\ 3 & 2.5 & 0 & 1.8 & 2.0 \\ 3.7 & 3.2 & 1.8 & 0 & 1.5 \\ 4.2 & 3.9 & 2.0 & 1.5 & 0 \end{bmatrix}$$

SL and CL produce the same hierarchy of clusterings at the levels given in the table.

Clustering	SL	CL
$\Re_0 = \{\{x_1\},\{x_2\},\{x_3\},\{x_4\},\{x_5\}\}$	0	0
$\Re_1 = \{\{x_1,x_2\},\{x_3\},\{x_4\},\{x_5\}\}$	1.2	1.2
$\Re_2 = \{\{x_1,x_2\},\{x_3\},\{x_4,x_5\}\}$	1.5	1.5
$\Re_3 = \{\{x_1,x_2\},\{x_3,x_4,x_5\}\}$	1.8	2.0
$\Re_4 = \{\{x_1,x_2,x_3,x_4,x_5\}\}$	2.5	4.2



Agglomerative graph theory based Clustering Algorithms

Remarks:

- SL poses the weakest possible graph condition (connectivity) for the formation of a cluster, while CL poses the strongest possible graph condition (completeness) for the formation of a cluster.
- A variety of graph theory-based algorithms, that lie between these two extremes result for various choices of $h(k)$.
  - For $k = 1$ all these algorithms **collapse** to the single link algorithm.
  - As k increases, the resulting subgraphs approach completeness.

Clustering algorithms based on the Minimum Spanning Tree (MST)

Definitions:

Spanning Tree: It is a connected graph (containing all the vertices of the graph), with no loops (only one path connects any two vertices).

Weight of a Spanning Tree: The sum of the weights of its edges (provided a weight has been assigned to each one of them).

Minimum Spanning Tree (MST): A spanning tree with the smallest weight among the spanning trees connecting all the vertices of the graph.

Agglomerative graph theory based Clustering Algorithms

Remarks:

- The MST has $N-1$ edges.
- When all the weights are different from each other, the MST is unique. Otherwise, it may not be unique.

➢ Employing the GTAS and substituting $g_{h(k)}(C_r, C_s)$ with

$$g(C_r, C_s) = min_{ij}\{w_{ij}: \boldsymbol{x}_i \in C_r, \boldsymbol{x}_j \in C_s\}$$

where $w_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j)$, we can determine the MST.

➢ **On the other hand,** a hierarchy of clusterings may be obtained by the MST as follows:

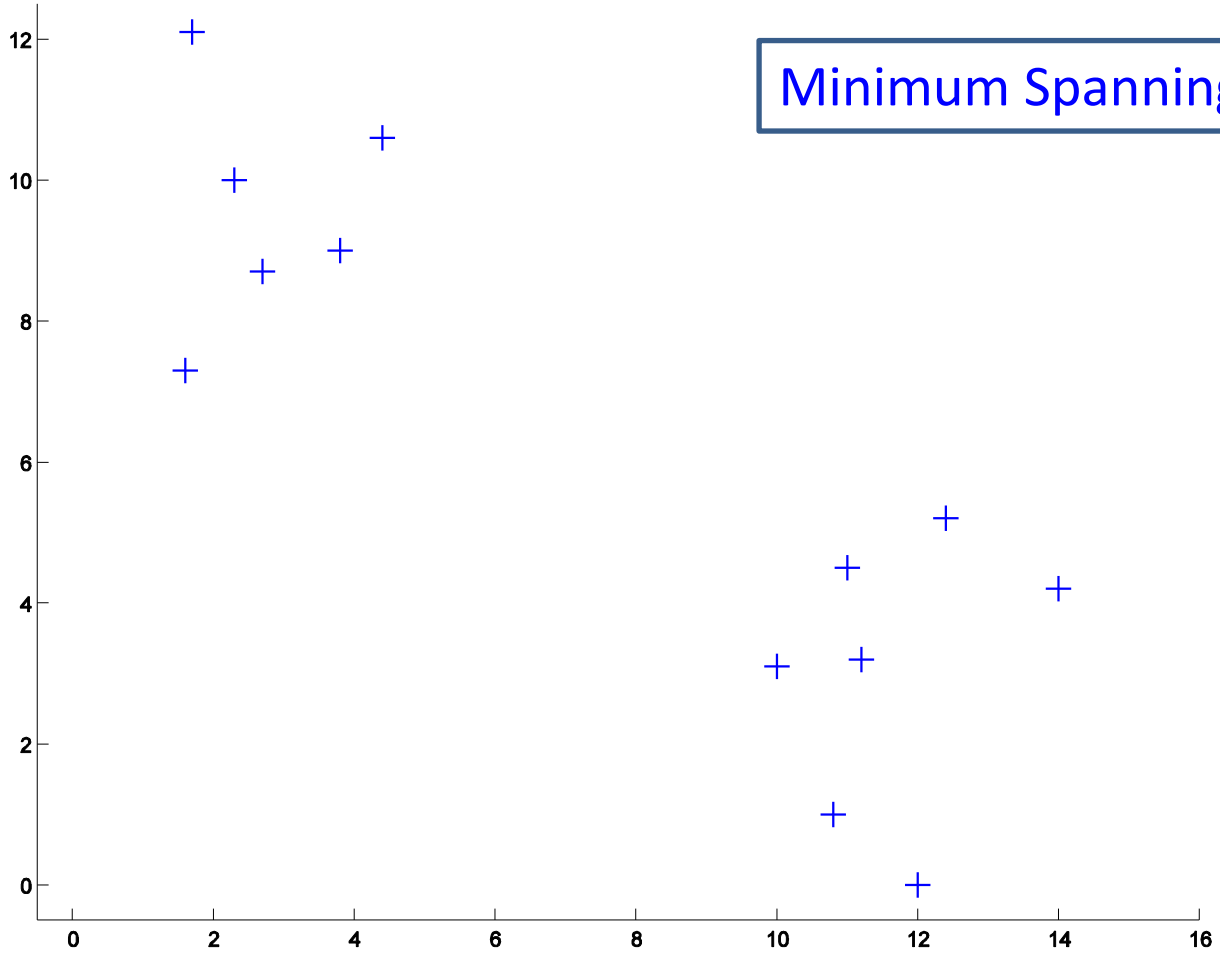
The clustering $\Re_t$ at the t −th level is the set of connected components of the MST, when only its t smallest weights are considered.

Remark:

The hierarchy produced by MST is the same with that produced by the single link algorithm, at least when all w_{ij}'s are different from each other.

Agglomerative graph theory based Clustering Algorithms

Example:

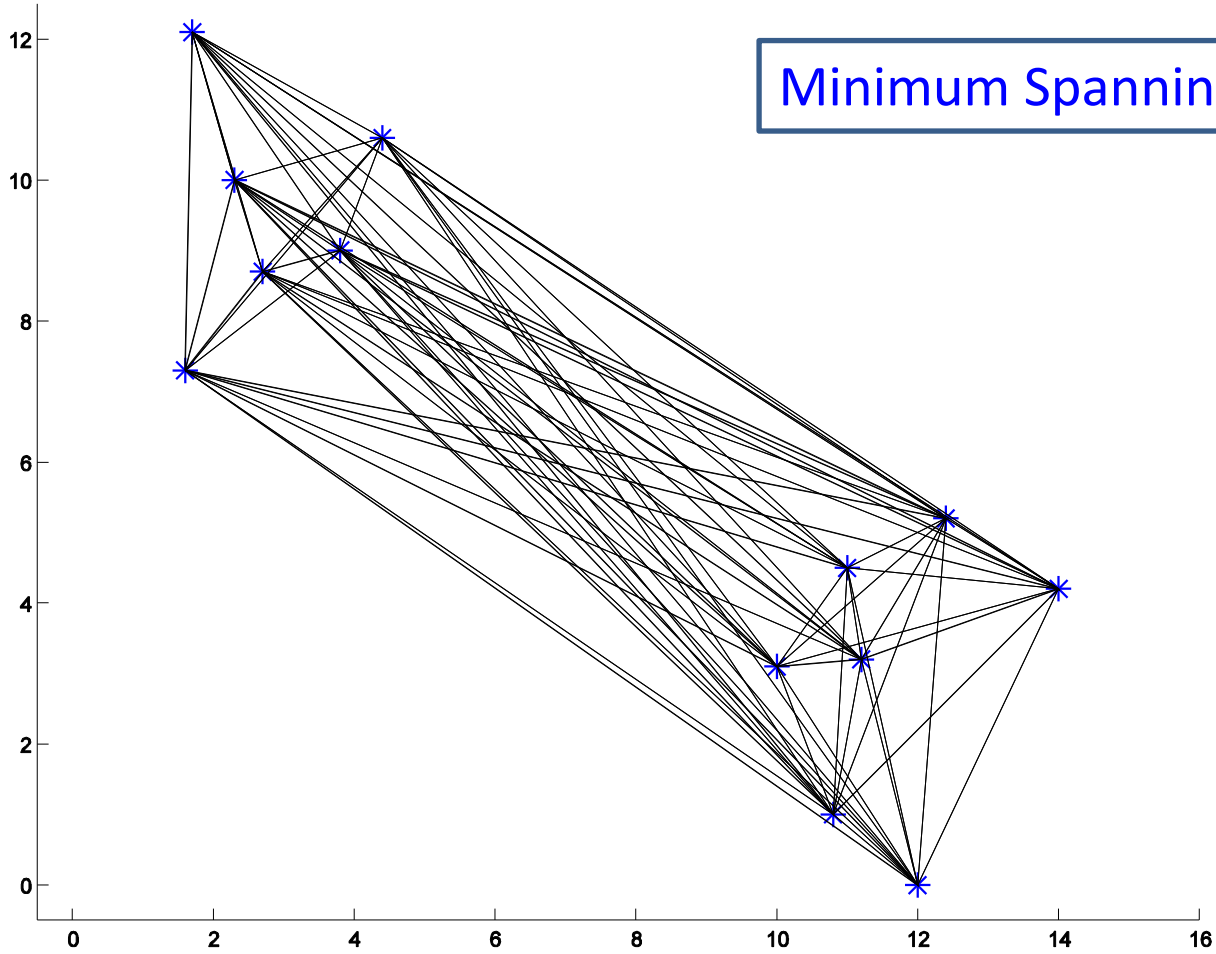


Minimum Spanning Tree (MST)

- **Define** a **complete** graph with vertices the data points and edges the segments connecting every pair of vertices.
- **Weight** each edge by the distance between its two end-points.
- **Define** the MST of the graph.

Agglomerative graph theory based Clustering Algorithms

Example:

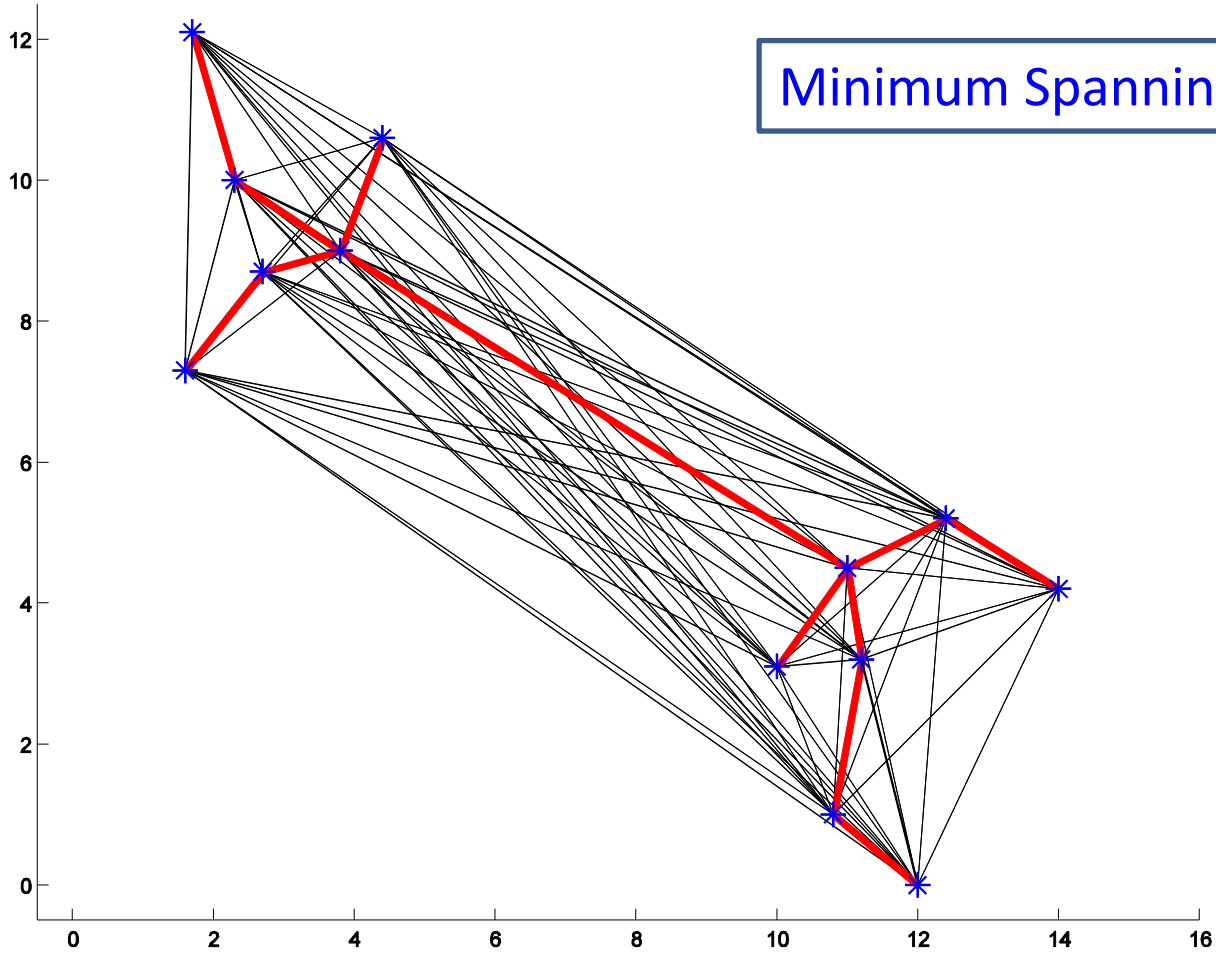


Minimum Spanning Tree (MST)

- **Define** a **complete** graph with vertices the data points and edges the segments connecting every pair of vertices.
- **Weight** each edge by the distance between its two end-points.
- **Define** the MST of the graph.

Agglomerative graph theory based Clustering Algorithms

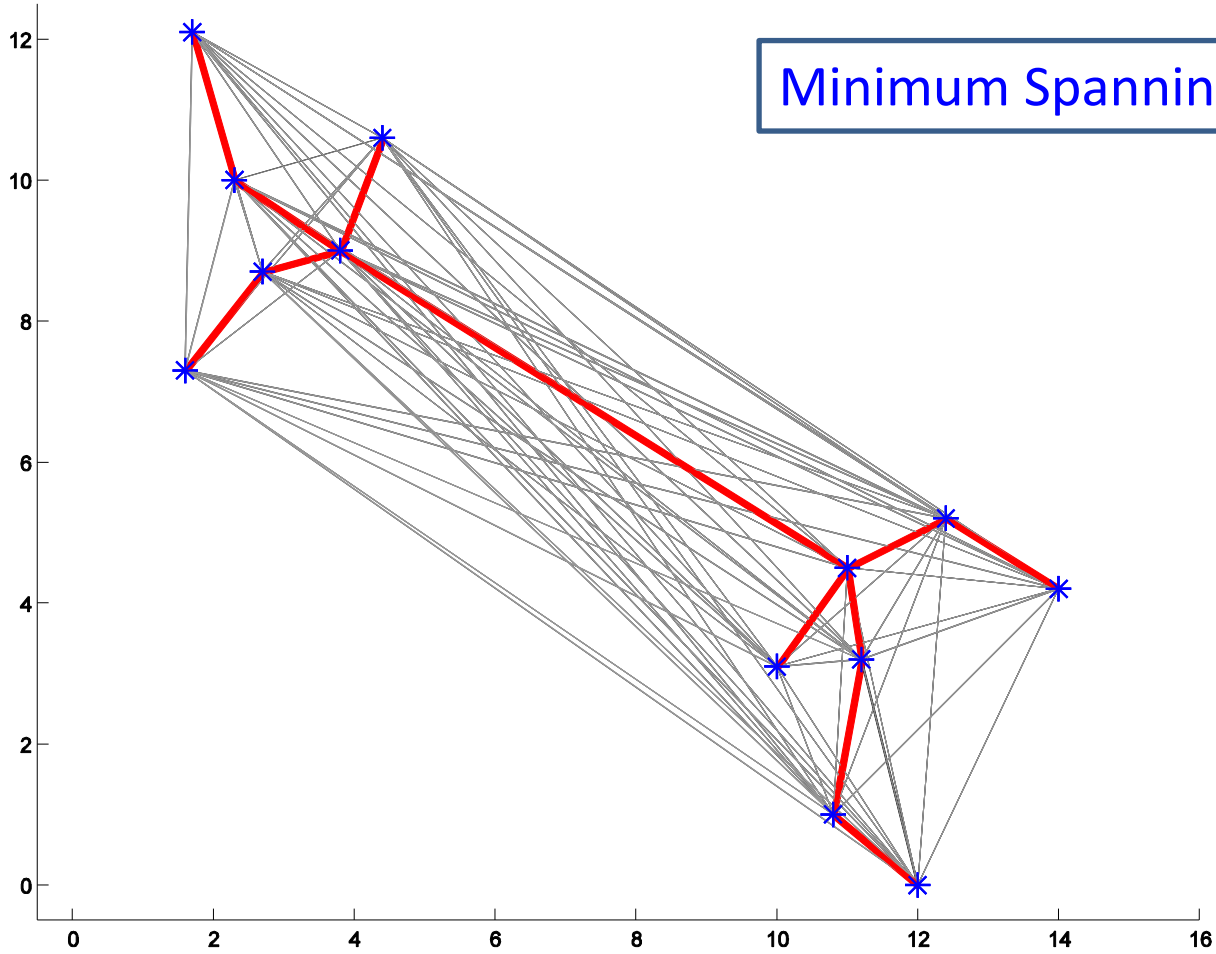
Example:



Minimum Spanning Tree (MST)

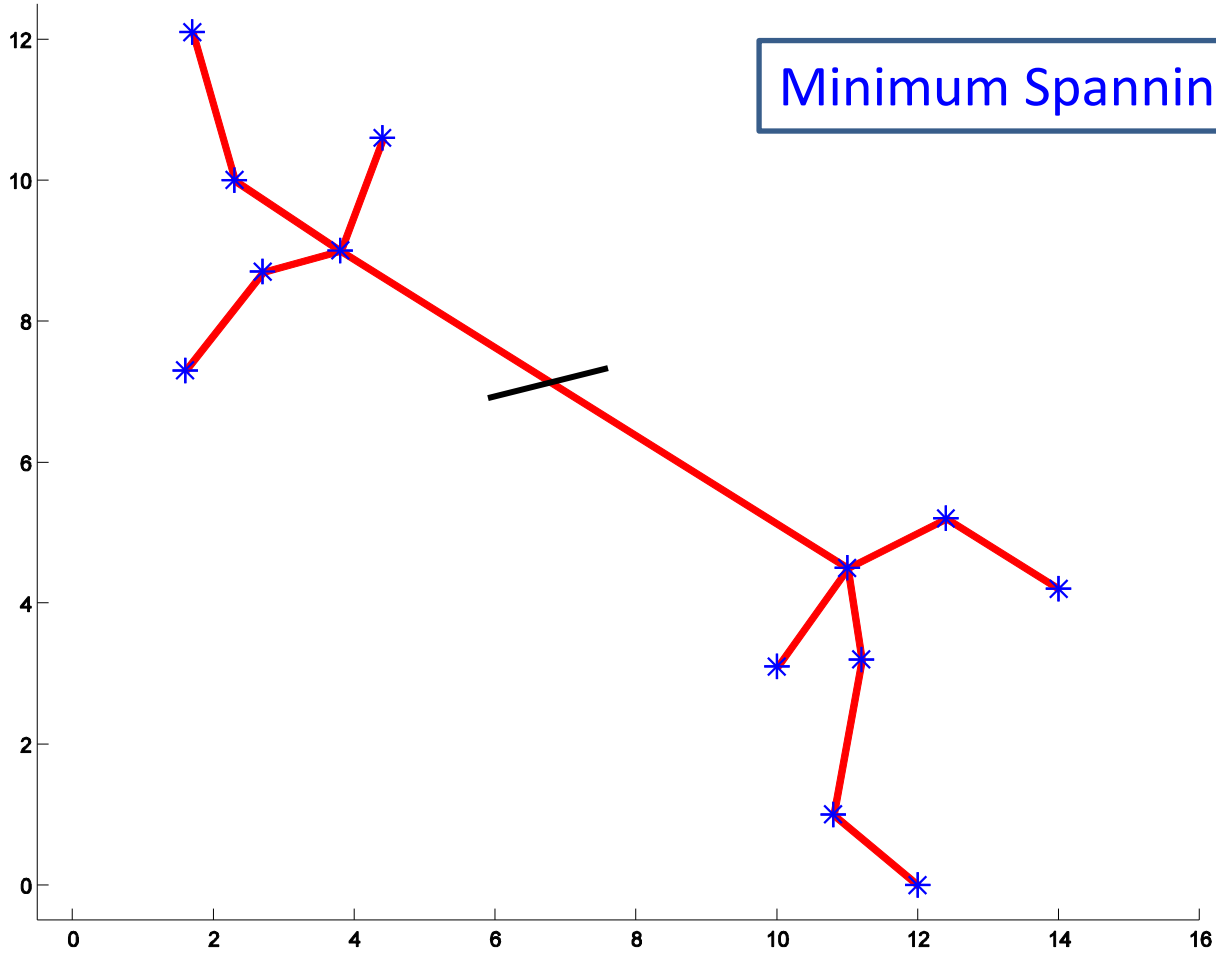
- **Define** a **complete** graph with vertices the data points and edges the segments connecting every pair of vertices.
- **Weight** each edge by the distance between its two end-points.
- **Define** the MST of the graph.

Agglomerative graph theory based Clustering Algorithms



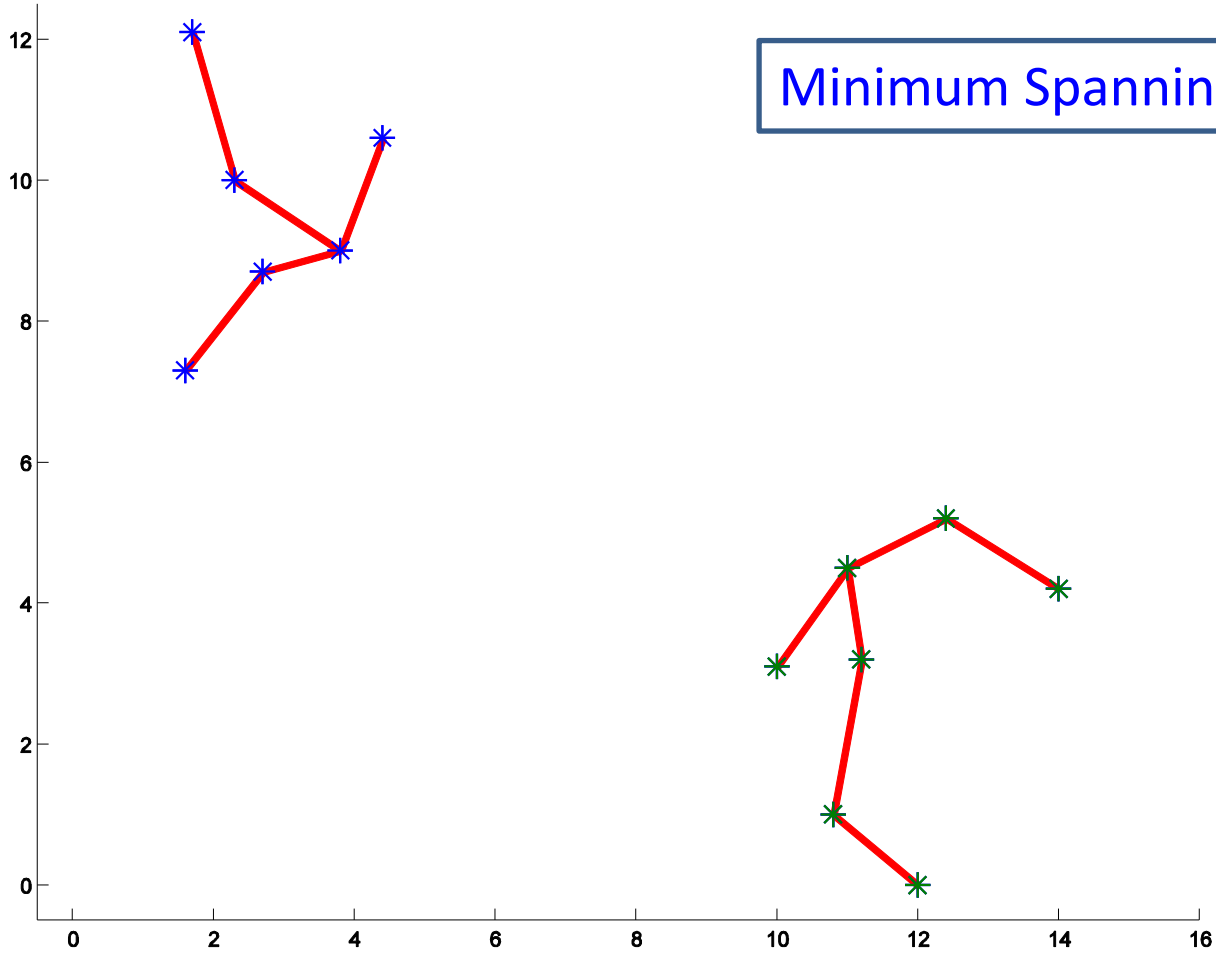
- **Define** a **complete** graph with vertices the data points and edges the segments connecting every pair of vertices.
- **Weight** each edge by the distance between its two end-points.
- **Define** the MST of the graph.

Agglomerative graph theory based Clustering Algorithms



- **Define** a **complete** graph with vertices the data points and edges the segments connecting every pair of vertices.
- **Weight** each edge by the distance between its two end-points.
- **Define** the MST of the graph.

Agglomerative graph theory based Clustering Algorithms



- **Define** a **complete** graph with vertices the data points and edges the segments connecting every pair of vertices.
- **Weight** each edge by the distance between its two end-points.
- **Define** the MST of the graph.
- **Retaining** the edges with the t **smallest weights**, the resulting connected components define the clusters of the $\mathfrak{R}_t$ clustering.