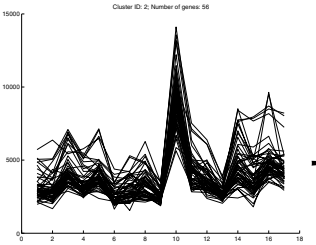

Chapter 7
Hierarchical Clustering Techniques

Hard clustering algorithms are subdivided into hierarchical algorithms and partitional algorithms. A partitional algorithm divides a data set into a single partition, whereas a hierarchical algorithm divides a data set into a sequence of nested partitions. As we mentioned in Chapter 1, hierarchical algorithms are subdivided into agglomerative hierarchical algorithms and divisive hierarchical algorithms (see Figure 1.5).

Agglomerative hierarchical clustering starts with every single object in a single cluster. Then it repeats merging the closest pair of clusters according to some similarity criteria until all of the data are in one cluster. There are some disadvantages for agglomerative hierarchical clustering, such as (a) data points that have been incorrectly grouped at an early stage cannot be reallocated and (b) different similarity measures for measuring the similarity between clusters may lead to different results.

If we treat agglomerative hierarchical clustering as a bottom-up clustering method, then divisive hierarchical clustering can be viewed as a top-down clustering method. Divisive hierarchical clustering starts with all objects in one cluster and repeats splitting large clusters into smaller pieces. Divisive hierarchical clustering has the same drawbacks as agglomerative hierarchical clustering. Figure 7.1 gives an intuitive example of agglomerative hierarchical clustering and divisive hierarchical clustering.

Hierarchical algorithms can be expressed in terms of either graph theory or matrix algebra (Jain and Dubes, 1988). A dendrogram, a special type of tree structure, is often used to visualize a hierarchical clustering. Figure 7.1 is an example of a dendrogram.

7.1 Representations of Hierarchical Clusterings

A hierarchical clustering can be represented by either a picture or a list of abstract symbols. A picture of a hierarchical clustering is much easier for humans to interpret. A list of abstract symbols of a hierarchical clustering may be used internally to improve the performance of the algorithm. In this section, some common representations of hierarchical clusterings are summarized.

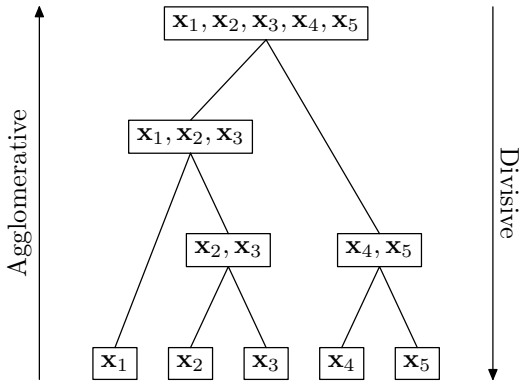

Figure 7.1. *Agglomerative hierarchical clustering and divisive hierarchical clustering.*

7.1.1 *n*-tree

A hierarchical clustering is generally represented by a tree diagram. An n-tree is a simple hierarchically nested tree diagram that can be used to represent a hierarchical clustering. Let $D = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ be a set of objects. Then an n-tree on D is defined to be a set $\mathcal{T}$ of subsets of D satisfying the following conditions (Bobisud and Bobisud, 1972; McMorris et al., 1983; Gordon, 1996):

1. $D \in \mathcal{T}$;
2. empty set $\Phi \in \mathcal{T}$;
3. $\{\mathbf{x}_i\} \in \mathcal{T}$ for all $i = 1, 2, \ldots, n$;
4. if $A, B \in \mathcal{T}$, then $A \cap B \in \{\Phi, A, B\}$.

A 5-tree is illustrated in Figure 7.2. The terminal nodes or leaves depicted by an open circle represent a single data point. The internal nodes depicted by a filled circle represent a group or cluster. n-trees are also referred to as nonranked trees (Murtagh, 1984b). If an n-tree has precisely $n-1$ internal nodes, then the tree is called a binary tree or a dichotomous tree.

Tree diagrams, such as n-trees and dendrograms (discussed later), contain many indeterminacies. For example, the order of the internal nodes and the order of leaves can be interchanged. Also, tree diagrams have many variations. For example, rotating the tree 90° gives a horizontal tree. Alternative properties of trees have been presented in (Hartigan, 1967) and (Constantinescu, 1966).

7.1.2 Dendrogram

A dendrogram is also called a valued tree (Gordon, 1996). A dendrogram is an n-tree in which each internal node is associated with a height satisfying the condition

$$h(A) \leq h(B) \Leftrightarrow A \subseteq B$$

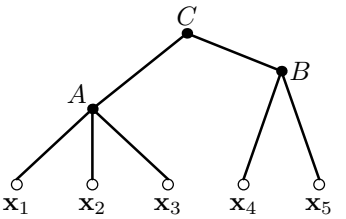

Figure 7.2. *A 5-tree.*

for all subsets of data points A and B if $A \cap B \neq \Phi$, where $h(A)$ and $h(B)$ denote the heights of A and B, respectively.

As an illustration, Figure 7.3 shows a dendrogram with five data points. The dotted lines indicate the heights of the internal nodes. For each pair of data points $(\mathbf{x}_i, \mathbf{x}_j)$, let h_{ij} be the height of the internal node specifying the smallest cluster to which both $\mathbf{x}_i$ and $\mathbf{x}_j$ belong. Then a small value of h_{ij} indicates a high similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. In the dendrogram given in Figure 7.3, for example, we have $h_{12} = 1$, $h_{23} = h_{13} = 3$, and $h_{14} = 4$.

The heights in the dendrogram satisfy the following ultrametric conditions (Johnson, 1967):

$$h_{ij} \leq \max\{h_{ik}, h_{jk}\} \quad \forall i, j, k \in \{1, 2, \ldots, n\}. \tag{7.1}$$

In fact, the ultrametric condition (7.1) is also a necessary and sufficient condition for a dendrogram (Gordon, 1987).

Mathematically, a dendrogram can be represented by a function $c : [0, \infty) \to E(D)$ that satisfies (Sibson, 1973)

$$c(h) \subseteq c(h') \text{ if } h \leq h',$$
$$c(h) \text{ is eventually in } D \times D,$$
$$c(h + \delta) = c(h) \text{ for some small } \delta > 0,$$

where D is a given data set and $E(D)$ is the set of equivalence relations on D. As an example, the function c given below contains the information of the dendrogram given in Figure 7.3:

$$c(h) = \begin{cases} \{(i,i) : i = 1,2,3,4,5\} & \text{if } 0 \leq h < 1, \\ \{(i,i) : i = 3,4,5\} \cup \\ \{(i,j) : i,j = 1,2\} & \text{if } 1 \leq h < 2, \\ \{(3,3)\} \cup \\ \{(i,j) : i,j = 1,2\} \cup \\ \{(i,j) : i,j = 4,5\} & \text{if } 2 \leq h < 3, \\ \{(i,j) : i,j = 4,5\} \cup \\ \{(i,j) : i,j = 1,2,3\} & \text{if } 3 \leq h < 4, \\ \{(i,j) : i,j = 1,2,3,4,5\} & \text{if } 4 \leq h. \end{cases} \tag{7.2}$$

Other characterizations of a dendrogram have been presented in (Johnson, 1967), (Jardine et al., 1967), and (Banfield, 1976). van Rijsbergen (1970) suggested an algorithm

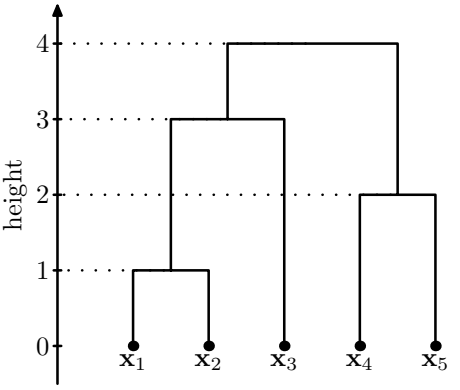

Figure 7.3. *A dendrogram of five data points.*

for finding the single-link dendrogram from the input dissimilarity matrix. Algorithms for plotting dendrograms have been discussed in (Rohlf, 1974), (Gower and Ross, 1969), and (Ross, 1969). Comparison of dendrograms has been discussed in (Sokal and Rohlf, 1962).

7.1.3 Banner

A banner (Rousseeuw, 1986) is a list of symbols and codes that represent a hierarchical structure. Banners can be constructed from dendrograms. In a banner, the heights in the dendrogram are represented on a horizontal axis. Each data point in the banner is assigned a line and a code that is repeated with a separator (such as "+") along the line until truncated at the right-hand margin. The presence of a symbol (such as "$*$") between two data points indicates that the two points are in the same group for this value of the height.

Figure 7.4 illustrates a banner that contains the information in the dendrogram given in Figure 7.3. In this banner, each data point is labeled by a two-number code. Alternative examples of banner representations are presented in (Kaufman and Rousseeuw, 1990, Chapter 6) and Gordon (1996).

7.1.4 Pointer Representation

A pointer representation (Sibson, 1973) is a pair of functions which contain information on a dendrogram. It is defined to be a pair of functions $\pi : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ and $\lambda : \pi(\{1, 2, \ldots, n\}) \to [0, \infty]$ that have the following properties:

$$\pi(n) = n, \quad \pi(i) > i \text{ for } i < n, \tag{7.3a}$$

$$\lambda(n) = \infty, \quad \lambda(\pi(i)) > \lambda(i) \text{ for } i < n, \tag{7.3b}$$

where n is the number of data points in D.

Given a dendrogram, the corresponding $\lambda(i)$ is the lowest level at which the ith object is no longer the last object in its cluster and $\pi(i)$ is the last object in the cluster that it joins.

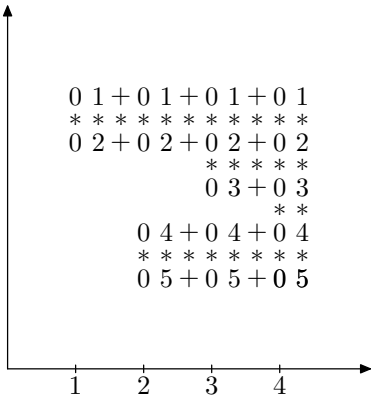

Figure 7.4. *A banner constructed from the dendrogram given in Figure 7.3.*

Mathematically, let c be the function that denotes a dendrogram. Then the corresponding pointer representation is defined by

$$\lambda(i) = \inf\{h : \exists j > i \text{ such that } (i, j) \in c(h)\},$$
$$\pi(i) = \max\{j : (i, j) \in c(\lambda(i))\}$$

for $i < n$.

The pair of functions λ and π illustrated in Table 7.1 is the pointer representation of the dendrogram given in Figure 7.3.

It can be shown that there is a one-to-one correspondence between dendrograms and pointer representations (Sibson, 1973). The pointer representation of a dendrogram allows a new object to be inserted in an efficient way. Usually, pointer representations are used internally in hierarchical algorithms in order to improve the performance. The pointer representation of a dendrogram is not helpful from a user's point of view; therefore, a so-called packed representation, which will be presented in the next section, is used for output.

Table 7.1. *The pointer representation corresponding to the dendrogram given in Figure 7.3.*

i	$\pi(i)$	$\lambda(i)$
1	2	1
2	3	3
3	5	4
4	5	2
5	5	∞

Table 7.2. *The packed representation corresponding to the pointer representation given in Table 7.1.*

i	$\tau(i)$	$\nu(i)$
1	1	1
2	2	3
3	3	4
4	4	2
5	5	∞

Table 7.3. *A packed representation of six objects.*

i	1	2	3	4	5	6
$\tau(i)$	4	1	3	5	2	6
$\nu(i)$	2	1.5	1	2.5	0.5	∞
$\lambda(i)$	1.5	0.5	1	2	2.5	∞
$\pi(i)$	5	6	5	5	6	6
$\lambda(\tau(i))$	2	1.5	1	2.5	0.5	∞
$\pi(\tau(i))$	5	5	5	6	6	6
$\tau^{-1}(\pi(\tau(i)))$	4	4	4	6	6	6

7.1.5 Packed Representation

Packed representations (Sibson, 1973) are developed in order to facilitate the output of dendrograms. A packed representation consists of two functions. Let λ, π be a pointer representation. Then the corresponding packed representation is defined as a pair of functions τ, ν (Sibson, 1973),

$$\tau : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}, \quad \nu : \{1, 2, \ldots, n\} \to [0, +\infty)$$

which satisfy the following conditions:

$$\begin{aligned}
&\tau \text{ is one-to-one and onto,}\\
&\tau^{-1}(\pi(\tau(i))) > i \text{ if } i < n,\\
&\nu(i) = \lambda(\tau(i)),\\
&\nu(j) \leq \nu(i) \text{ if } i \leq j < \tau^{-1}(\pi(\tau(i))).
\end{aligned}$$

The packed representation defined above determines a dendrogram uniquely. In fact, the order of the objects in the dendrogram is specified by the function τ, i.e., the index of the object in position i is $\tau(i)$. Table 7.2 gives the packed representation corresponding to the pointer representation given in Table 7.1.

Another example of a packed representation is illustrated in Table 7.3. The dendrogram determined by this packed representation is shown in Figure 7.5.

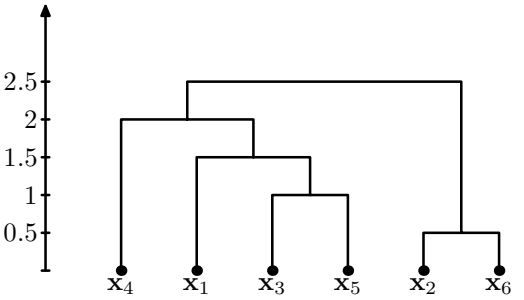

Figure 7.5. *The dendrogram determined by the packed representation given in Table 7.3.*

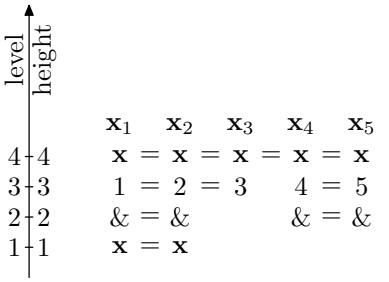

Figure 7.6. *An icicle plot corresponding to the dendrogram given in Figure 7.3.*

7.1.6 Icicle Plot

An icicle plot, proposed by Kruskal and Landwehr (1983), is another method for presenting a hierarchical clustering. It can be constructed from a dendrogram. The major advantage of an icicle plot is that it is easy to read off which objects are in a cluster during a live process of data analysis.

In an icicle plot, the height and the hierarchical level are represented along the vertical axis; each object is assigned a vertical line and labeled by a code that is repeated with separators (such as "&") along the line from top to bottom until truncated at the level where it first joins a cluster, and objects in the same cluster are joined by the symbol "=" between two objects.

The list of symbols in Figure 7.6 is an icicle plot corresponding to the dendrogram given in Figure 7.3. Each object in this icicle plot is labeled by its name.

7.1.7 Other Representations

Other representations of hierarchical structures have been presented in (Sokal and Sneath, 1973), (Friedman and Rafsky, 1981), (Everitt and Nicholls, 1975), (Wirth et al., 1966), and (Hartigan and Wong, 1979), such as skyline plots (Wirth et al., 1966), silhouette plots (Rousseeuw, 1987), loop plots (see Figure 7.7) (Kruskal and Landwehr, 1983), and three-

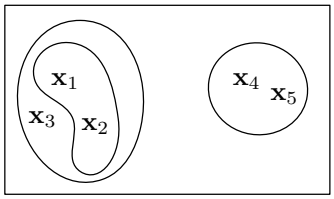

Figure 7.7. *A loop plot corresponding to the dendrogram given in Figure 7.3.*

dimensional plots (Kruskal and Landwehr, 1983). It seems, however, that these representations are only suitable for representing small data sets.

7.2 Agglomerative Hierarchical Methods

According to different distance measures between groups, agglomerative hierarchical methods can be subdivided into single-link methods, complete link methods, etc. Some commonly used hierarchical methods are given in Figure 7.8. The single, complete, average, and weighted average linkage methods are also referred to as graph methods, while Ward's method, the centroid method, and the median method are referred to as geometric methods (Murtagh, 1983), since in graph methods a cluster can be represented by a subgraph or interconnected points and in geometric methods a cluster can be represented by a center point.

Murtagh (1983) gives a survey for hierarchical clustering algorithms, especially for agglomerative hierarchical clustering algorithms. The performance of hierarchical clustering algorithms can be improved by incorporating efficient nearest neighbor searching algorithms into the clustering algorithms. For hierarchical methods, the storage requirements are reduced if each cluster is represented by a center point or a set of points, since

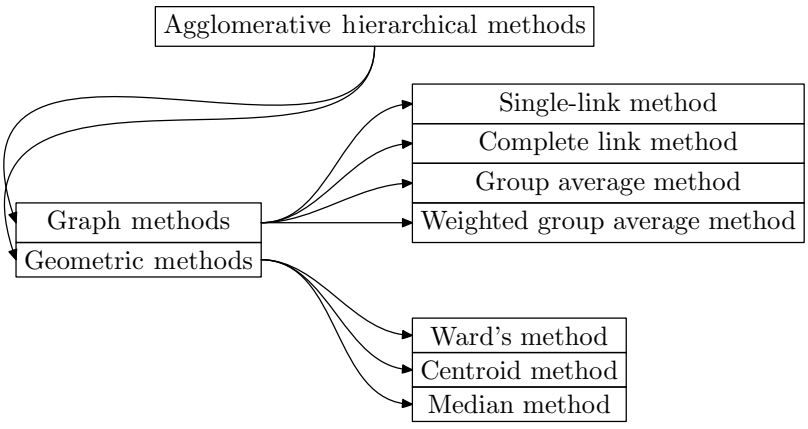

Figure 7.8. *Some commonly used hierarchical methods.*

Table 7.4. *The cluster centers agglomerated from two clusters and the dissimilarities between two cluster centers for geometric hierarchical methods, where $\mu(C)$ denotes the center of cluster C.*

Hierarchical Method	$\mu(C_i \cup C_j)$	*Dissimilarity between C_i and C_j*
Median	$\frac{\mu(C_i)+\mu(C_j)}{2}$	$\|\mu(C_i) - \mu(C_j)\|^2$
Centroid	$\frac{\|C_i\|\mu(C_i)+\|C_j\|\mu(C_j)}{\|C_i\|+\|C_j\|}$	$\|\mu(C_i) - \mu(C_j)\|^2$
Ward's	$\frac{\|C_i\|\mu(C_i)+\|C_j\|\mu(C_j)}{\|C_i\|+\|C_j\|}$	$\frac{\|C_i\|\|C_j\|}{\|C_i\|+\|C_j\|}\|\mu(C_i) - \mu(C_j)\|^2$

$O(n^2)$ storage is required for a dissimilarity matrix, where n is the number of data points. Also for geometric hierarchical methods, the representative point of a cluster can be derived directly from that of the two clusters that form the cluster (see Table 7.4).

In agglomerative hierarchical clustering algorithms, the Lance-Williams formula (cf. Section 6.8) is used to calculate the dissimilarity between a cluster and a cluster formed by merging two other clusters. The single-link and complete link hierarchical clustering algorithms induce a metric on the data known as the ultrametric (Johnson, 1967). But the hierarchical structures produced by other clustering algorithms that use the Lance-Williams recurrence formula may violate the ultrametric inequality (Milligan, 1979).

The distances $D(C_k, C_i \cup C_j)$ are said to increase monotonically if $D(C_k, C_i \cup C_j) \geq D(C_i, C_j)$ at each level in the hierarchy. If an algorithm produces a monotonic hierarchy, then the algorithm induces a type of distance metric known as the ultrametric (Milligan, 1979). The centroid method and median method are examples of hierarchical algorithms that do not produce monotonic hierarchies.

Milligan (1979) has shown that the hierarchical clustering strategy $(\alpha_1, \alpha_2, \beta, \gamma)$ based on the Lance-Williams recurrence formula (Lance and Williams, 1967b) is monotonic, i.e.,

$$D(C_k, C_i \cup C_j) \geq D(C_i, C_j) \quad \forall i, j, k,$$

if the following conditions are satisfied:

1. $\gamma \geq 0 \vee (\gamma < 0 \wedge |\gamma| \leq \alpha_1, \alpha_2)$,
2. $\min\{\alpha_1, \alpha_2\} \geq 0$,
3. $\alpha_1 + \alpha_2 + \beta \geq 1$.

Also, Batagelj (1981) gives a necessary and sufficient condition for the hierarchical clustering strategy $(\alpha_1, \alpha_2, \beta, \gamma)$ to be monotonic:

1. $\gamma \geq -\min\{\alpha_1, \alpha_2\}$,
2. $\alpha_1 + \alpha_2 \geq 0$,
3. $\alpha_1 + \alpha_2 + \beta \geq 1$.

7.2.1 The Single-link Method

The single-link method is one of the simplest hierarchical clustering methods. It was first introduced by Florek et al. (1951) and then independently by McQuitty (1957) and Sneath (1957). The single-link method is also known by other names, such as the nearest neighbor method, the minimum method, and the connectedness method (Rohlf, 1982). The single-link method is invariant under monotone transformations (such as the logarithmic transformation) of the original data (Johnson, 1967).

It employs the nearest neighbor distance (cf. Section 6.8) to measure the dissimilarity between two groups. Let C_i, C_j, and C_k be three groups of data points. Then the distance between C_k and $C_i \cup C_j$ can be obtained from the Lance-Williams formula as follows:

$$\begin{aligned} & D(C_k, C_i \cup C_j) \\ &= \frac{1}{2}D(C_k, C_i) + \frac{1}{2}D(C_k, C_j) - \frac{1}{2}|D(C_k, C_i) - D(C_k, C_j)| \\ &= \min\{D(C_k, C_i), D(C_k, C_j)\}, \end{aligned} \tag{7.4}$$

where $D(\cdot, \cdot)$ is a distance between two clusters.

From equation (7.4), it is not difficult to verify that

$$D(C, C') = \min_{\mathbf{x}\in C, \mathbf{y}\in C'} d(\mathbf{x}, \mathbf{y}),$$

where C and C' are two nonempty, nonoverlapping clusters and $d(\cdot, \cdot)$ is the distance function by which the dissimilarity matrix is computed.

Rohlf (1982) has classified single-link algorithms into five different types:

1. connectedness algorithms,
2. algorithms based on an ultrametric transformation,
3. probability density estimation algorithms,
4. agglomerative algorithms,
5. algorithms based on the minimum spanning tree.

The connectedness algorithms are based on graph theory. In a connectedness algorithm, the data points are represented as vertices in a graph: a pair (i, j) of vertices are connected with an edge if and only if the distance between data points i and j $d_{ij} \leq \Delta$. The single-link clusters at level Δ correspond to the connected subgraphs of the graph. The connectedness algorithms require a considerable amount of computational effort. van Groenewoud and Ihm (1974) presented such an algorithm, whose total time complexity is $O(n^5)$, where n is the size of the data set.

More single-link algorithms will be presented in later sections of this chapter. What follows is a simple example that illustrates the idea of the single-link algorithm.

For the data set given in Figure 7.9, for example, the dissimilarity matrix computed using the Euclidean distance is described in Table 7.5. If a single-link hierarchical clustering algorithm is applied to this data set, then $\mathbf{x}_1$ and $\mathbf{x}_2$ will be agglomerated to form a big cluster

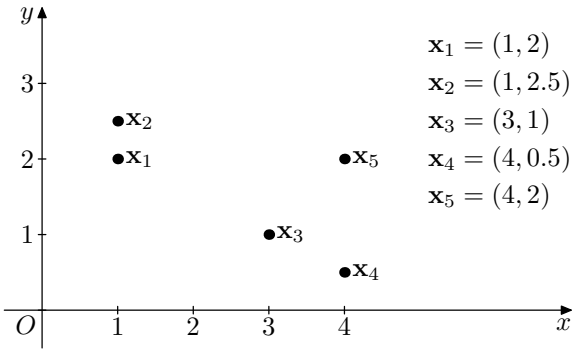

Figure 7.9. *A two-dimensional data set with five data points.*

Table 7.5. *The dissimilarity matrix of the data set given in Figure 7.9. The entry* (i, j) *in the matrix is the Euclidean distance between* $\mathbf{x}_i$ *and* $\mathbf{x}_j$.

	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\mathbf{x}_1$	0	0.5	2.24	3.35	3
$\mathbf{x}_2$	0.5	0	2.5	3.61	3.04
$\mathbf{x}_3$	2.24	2.5	0	1.12	1.41
$\mathbf{x}_4$	3.35	3.61	1.12	0	1.5
$\mathbf{x}_5$	3	3.04	1.41	1.5	0

at the first stage of the algorithm, since they have the least distance in the dissimilarity matrix. The distance between $\{\mathbf{x}_1, \mathbf{x}_2\}$ and $\mathbf{x}_3$, $\mathbf{x}_4$, and $\mathbf{x}_5$ now becomes

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3) = \min\{d(\mathbf{x}_1, \mathbf{x}_3), d(\mathbf{x}_2, \mathbf{x}_3)\} = 2.24,$$
$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4) = \min\{d(\mathbf{x}_1, \mathbf{x}_4), d(\mathbf{x}_2, \mathbf{x}_4)\} = 3.35,$$
$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_5) = \min\{d(\mathbf{x}_1, \mathbf{x}_5), d(\mathbf{x}_2, \mathbf{x}_5)\} = 3,$$

which can also be obtained from the formula given in (7.4). After $\mathbf{x}_1$ and $\mathbf{x}_2$ are merged, the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.24	3.35	3
$\mathbf{x}_3$	2.24	0	1.12	1.41
$\mathbf{x}_4$	3.35	1.12	0	1.5
$\mathbf{x}_5$	3	1.41	1.5	0

At the second stage of the algorithm, $\mathbf{x}_3$ and $\mathbf{x}_4$ will be merged, since they have the least distance. Then the distances between the group $\{\mathbf{x}_3, \mathbf{x}_4\}$ and the remaining groups

become

$$\begin{aligned} &D(\{\mathbf{x}_3, \mathbf{x}_4\}, \{\mathbf{x}_1, \mathbf{x}_2\}) \\ &= \min\{d(\mathbf{x}_1, \mathbf{x}_3), d(\mathbf{x}_2, \mathbf{x}_3), d(\mathbf{x}_1, \mathbf{x}_4), d(\mathbf{x}_2, \mathbf{x}_4)\} \\ &= \min\{D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3), D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4)\} = 2.24 \end{aligned}$$

and

$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \mathbf{x}_5) = \min\{d(\mathbf{x}_3, \mathbf{x}_5), d(\mathbf{x}_4, \mathbf{x}_5)\} = 1.41.$$

After $\mathbf{x}_3$ and $\mathbf{x}_4$ are merged, the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4\}$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.24	3
$\{\mathbf{x}_3, \mathbf{x}_4\}$	2.24	0	1.41
$\mathbf{x}_5$	3	1.41	0

At the third stage of the algorithm, $\{\mathbf{x}_3, \mathbf{x}_4\}$ and $\mathbf{x}_5$ will be merged. The dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.24
$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$	2.24	0

At the fourth stage, all the data points are merged into a single cluster. The dendrogram of this clustering is shown in Figure 7.10.

7.2.2 The Complete Link Method

Unlike the single-link method, the complete link method uses the farthest neighbor distance (cf. Section 6.8) to measure the dissimilarity between two groups. The complete link

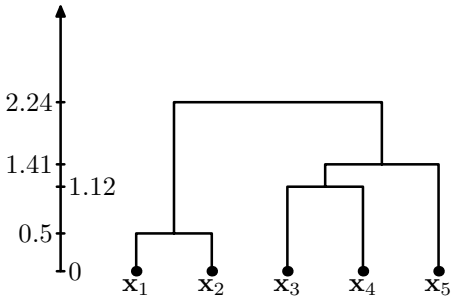

Figure 7.10. *The dendrogram produced by applying the single-link method to the data set given in Figure 7.9.*

method is also invariant under monotone transformations (Johnson, 1967). Let C_i, C_j, and C_k be three groups of data points. Then the distance between C_k and $C_i \cup C_j$ can be obtained from the Lance-Williams formula as follows:

$$\begin{aligned} & D(C_k, C_i \cup C_j) \\ &= \frac{1}{2}D(C_k, C_i) + \frac{1}{2}D(C_k, C_j) + \frac{1}{2}|D(C_k, C_i) - D(C_k, C_j)| \\ &= \max\{D(C_k, C_i), D(C_k, C_j)\}, \end{aligned} \tag{7.5}$$

where $D(\cdot, \cdot)$ is a distance between two clusters.

The distance defined in equation (7.5) has the following property:

$$D(C, C') = \max_{\mathbf{x}\in C, \mathbf{y}\in C'} d(\mathbf{x}, \mathbf{y}),$$

where C and C' are two nonempty, nonoverlapping clusters and $d(\cdot, \cdot)$ is the distance function by which the dissimilarity matrix is computed.

Applying the complete link method to the dissimilarity matrix given in Table 7.5, at the first stage we merge $\mathbf{x}_1$ and $\mathbf{x}_2$. The distances between the group $\{\mathbf{x}_1, \mathbf{x}_2\}$ and the remaining three points are updated as

$$\begin{aligned} D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3) &= \max\{d(\mathbf{x}_1, \mathbf{x}_3), d(\mathbf{x}_2, \mathbf{x}_3)\} = 2.5, \\ D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4) &= \max\{d(\mathbf{x}_1, \mathbf{x}_4), d(\mathbf{x}_2, \mathbf{x}_4)\} = 3.61, \\ D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_5) &= \max\{d(\mathbf{x}_1, \mathbf{x}_5), d(\mathbf{x}_2, \mathbf{x}_5)\} = 3.04, \end{aligned}$$

which can also be obtained from the formula given in equation (7.5). After $\mathbf{x}_1$ and $\mathbf{x}_2$ are merged at the first stage of the algorithm, the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.5	3.61	3.04
$\mathbf{x}_3$	2.5	0	1.12	1.41
$\mathbf{x}_4$	3.61	1.12	0	1.5
$\mathbf{x}_5$	3.04	1.41	1.5	0

Again, at the second stage of the algorithm, $\mathbf{x}_3$ and $\mathbf{x}_4$ will be merged, since they have the least distance between them. After $\mathbf{x}_3$ and $\mathbf{x}_4$ are merged, the distances between the group $\{\mathbf{x}_3, \mathbf{x}_4\}$ and the remaining groups are updated as

$$\begin{aligned} & D(\{\mathbf{x}_3, \mathbf{x}_4\}, \{\mathbf{x}_1, \mathbf{x}_2\}) \\ &= \max\{d(\mathbf{x}_1, \mathbf{x}_3), d(\mathbf{x}_2, \mathbf{x}_3), d(\mathbf{x}_1, \mathbf{x}_4), d(\mathbf{x}_2, \mathbf{x}_4)\} \\ &= \max\{D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3), D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4)\} = 3.61 \end{aligned}$$

and

$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \mathbf{x}_5) = \max\{d(\mathbf{x}_3, \mathbf{x}_5), d(\mathbf{x}_4, \mathbf{x}_5)\} = 1.5.$$

After $\mathbf{x}_3$ and $\mathbf{x}_4$ are merged, the dissimilarity matrix becomes

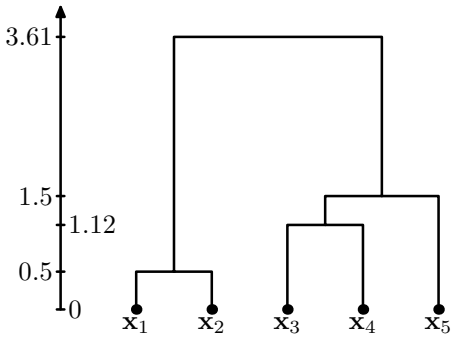

Figure 7.11. *The dendrogram produced by applying the complete link method to the data set given in Figure 7.9.*

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4\}$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	3.61	3.04
$\{\mathbf{x}_3, \mathbf{x}_4\}$	3.61	0	1.5
$\mathbf{x}_5$	3.04	1.5	0

At the third stage of the algorithm, $\{\mathbf{x}_3, \mathbf{x}_4\}$ and $\mathbf{x}_5$ must be merged, since they have the least distance. After $\mathbf{x}_4$ and $\mathbf{x}_5$ are merged, the distance between the two groups is

$$\begin{aligned} &D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) \\ &= \max\{d_{13}, d_{14}, d_{15}, d_{23}, d_{24}, d_{25}\} \\ &= \max\{D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4\}), D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_5)\} \\ &= 3.61, \end{aligned}$$

where $d_{ij} = d(\mathbf{x}_i, \mathbf{x}_j)$ for $i = 1, 2$ and $j = 3, 4, 5$, and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	3.61
$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$	3.61	0

The dendrogram produced by the complete link method is shown in Figure 7.11, which is the same as the dendrogram produced by the single-link method except for the heights.

7.2.3 The Group Average Method

The group average method is also referred as UPGMA, which stands for "unweighted pair group method using arithmetic averages" (Jain and Dubes, 1988). In the group average method, the distance between two groups is defined as the average of the distances between all possible pairs of data points that are made up of one data point from each group. Let C_i, C_j, and C_k be three groups of data points. Then the distance between C_k and $C_i \cup C_j$ can

be obtained from the Lance-Williams formula as follows:

$$\begin{aligned} &D(C_k, C_i \cup C_j) \\ &= \frac{|C_i|}{|C_i| + |C_j|} D(C_k, C_i) + \frac{|C_j|}{|C_i| + |C_j|} D(C_k, C_j), \end{aligned} \tag{7.6}$$

where $D(\cdot, \cdot)$ is a distance between two clusters.

Let C and C' be two nonempty, nonoverlapping clusters. Then in the group average method, we have

$$D(C, C') = \frac{1}{|C||C'|} \sum_{\mathbf{x} \in C, \mathbf{y} \in C'} d(\mathbf{x}, \mathbf{y}), \tag{7.7}$$

where $d(\cdot, \cdot)$ is the distance function by which the dissimilarity matrix is computed.

In fact, let C_1, C_2, and C_3 be three nonempty, mutually nonoverlapping clusters, and assume

$$D(C_i, C_j) = \frac{1}{n_i n_j} \Sigma(C_i, C_j), \quad 1 \le i < j \le 3, \tag{7.8}$$

where $n_i = |C_i|$, $n_j = |C_j|$, and $\Sigma(C_i, C_j)$ is the total between-clusters distance of C_i and C_j, that is,

$$\Sigma(C_i, C_j) = \sum_{\mathbf{x} \in C_i, \mathbf{y} \in C_j} d(\mathbf{x}, \mathbf{y}).$$

Now from equations (7.6) and (7.8), we have

$$\begin{aligned} &D(C_1, C_2 \cup C_3) \\ &= \frac{n_2}{n_2 + n_3} D(C_1, C_2) + \frac{n_3}{n_2 + n_3} D(C_1, C_3) \\ &= \frac{n_2}{n_2 + n_3} \cdot \frac{1}{n_1 n_2} \Sigma(C_1, C_2) + \frac{n_3}{n_2 + n_3} \cdot \frac{1}{n_1 n_3} \Sigma(C_1, C_3) \\ &= \frac{1}{n_1(n_2 + n_3)} \Sigma(C_1, C_2 \cup C_3), \end{aligned}$$

since $\Sigma(C_1, C_2) + \Sigma(C_1, C_3) = \Sigma(C_1, C_2 \cup C_3)$. This verifies equation (7.7).

Applying the group average method to the data set given in Figure 7.9, we note that again the first stage is to merge $\mathbf{x}_1$ and $\mathbf{x}_2$. After $\mathbf{x}_1$ and $\mathbf{x}_2$ are merged, the distances between $\{\mathbf{x}_1, \mathbf{x}_2\}$ and the remaining three data points become

$$\begin{aligned} D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3) &= \frac{1}{2} d(\mathbf{x}_1, \mathbf{x}_3) + \frac{1}{2} d(\mathbf{x}_2, \mathbf{x}_3) = 2.37, \\ D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4) &= \frac{1}{2} d(\mathbf{x}_1, \mathbf{x}_4) + \frac{1}{2} d(\mathbf{x}_2, \mathbf{x}_4) = 3.48, \\ D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_5) &= \frac{1}{2} d(\mathbf{x}_1, \mathbf{x}_5) + \frac{1}{2} d(\mathbf{x}_2, \mathbf{x}_5) = 3.02, \end{aligned}$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.37	3.48	3.02
$\mathbf{x}_3$	2.37	0	1.12	1.41
$\mathbf{x}_4$	3.48	1.12	0	1.5
$\mathbf{x}_5$	3.02	1.41	1.5	0

Again, at the second stage of the algorithm, $\mathbf{x}_4$ and $\mathbf{x}_3$ will be merged. The distances between $\{\mathbf{x}_3, \mathbf{x}_4\}$ and the other clusters become

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4\}) = \frac{1}{2}D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4) + \frac{1}{2}D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3) = 2.93,$$
$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \mathbf{x}_5) = \frac{1}{2}d(\mathbf{x}_3, \mathbf{x}_5) + \frac{1}{2}d(\mathbf{x}_4, \mathbf{x}_5) = 1.46.$$

After $\mathbf{x}_3$ and $\mathbf{x}_4$ are merged into one cluster, the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4\}$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.93	3.02
$\{\mathbf{x}_3, \mathbf{x}_4\}$	2.93	0	1.46
$\mathbf{x}_5$	3.02	1.46	0

At the third stage of the algorithm, $\{\mathbf{x}_3, \mathbf{x}_4\}$ and $\mathbf{x}_5$ must be merged, since they have the least distance. Then the distance between $\{\mathbf{x}_1, \mathbf{x}_2\}$ and $\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$ becomes

$$\begin{aligned} &D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) \\ &= \frac{2}{3}D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4\}) + \frac{1}{3}D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_5) \\ &= 2.96. \end{aligned}$$

Hence, the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.96
$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$	2.96	0

The distances can also be calculated by equation (7.7). In the last stage, for example, the distance between $\{\mathbf{x}_1, \mathbf{x}_2\}$ and $\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$ can be computed as

$$\begin{aligned} &D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) \\ &= \frac{1}{6}(d_{13} + d_{14} + d_{15} + d_{23} + d_{24} + d_{25}) = 2.96, \end{aligned}$$

where $d_{ij} = d(\mathbf{x}_i, \mathbf{x}_j)$, i.e., the (i, j)th entry of the dissimilarity matrix.

The dendrogram of this clustering is shown in Figure 7.12.

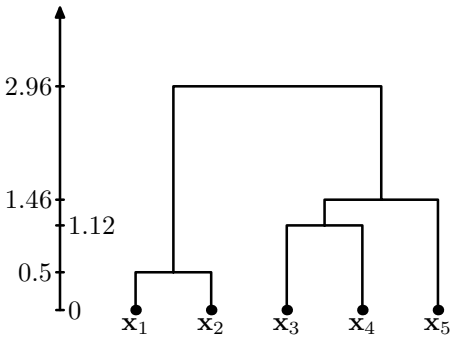

Figure 7.12. *The dendrogram produced by applying the group average method to the data set given in Figure 7.9.*

7.2.4 The Weighted Group Average Method

The weighted group average method is also referred to as the "weighted pair group method using arithmetic average" (Jain and Dubes, 1988). Using the Lance-Williams formula, the distance between clusters is

$$D(C_k, C_i \cup C_j) = \frac{1}{2}D(C_k, C_i) + \frac{1}{2}D(C_k, C_j),$$

where C_k, C_i, and C_j are three clusters in one level of clustering.

Applying the weighted group average method to the five-point data set given in Figure 7.9, the first stage is the same as in the other methods, i.e., we merge $\mathbf{x}_1$ and $\mathbf{x}_2$. After $\mathbf{x}_1$ and $\mathbf{x}_2$ are merged, the distances between clusters are updated as

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3) = \frac{1}{2}(d(\mathbf{x}_1, \mathbf{x}_3) + d(\mathbf{x}_2, \mathbf{x}_3)) = 2.37,$$
$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4) = \frac{1}{2}(d(\mathbf{x}_1, \mathbf{x}_4) + d(\mathbf{x}_2, \mathbf{x}_4)) = 3.48,$$
$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_5) = \frac{1}{2}(d(\mathbf{x}_1, \mathbf{x}_5) + d(\mathbf{x}_2, \mathbf{x}_5)) = 3.02,$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.37	3.48	3.02
$\mathbf{x}_3$	2.37	0	1.12	1.41
$\mathbf{x}_4$	3.48	1.12	0	1.5
$\mathbf{x}_5$	3.02	1.41	1.5	0

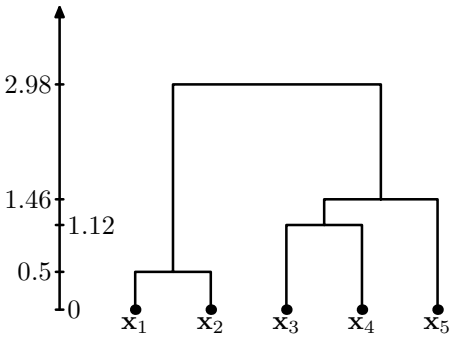

Figure 7.13. *The dendrogram produced by applying the weighted group average method to the data set given in Figure 7.9.*

At the second stage of this method, $\mathbf{x}_3$ and $\mathbf{x}_4$ will be merged. After $\mathbf{x}_3$ and $\mathbf{x}_4$ are merged, the distances between clusters are updated as

$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \{\mathbf{x}_1, \mathbf{x}_2\}) = \frac{1}{2}(2.37 + 3.48) = 2.93,$$
$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \mathbf{x}_5) = \frac{1}{2}(1.41 + 1.5) = 1.46,$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4\}$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.93	3.02
$\{\mathbf{x}_3, \mathbf{x}_4\}$	2.93	0	1.46
$\mathbf{x}_5$	3.02	1.46	0

Clusters $\{\mathbf{x}_3, \mathbf{x}_4\}$ and $\mathbf{x}_5$ will be merged at the third stage of this method. The distance is updated as

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) = \frac{1}{2}(2.93 + 3.02) = 2.98,$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.98
$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$	2.98	0

The whole process of this clustering can be represented by the dendrogram shown in Figure 7.13.

7.2.5 The Centroid Method

The centroid method is also referred to as the "unweighted pair group method using centroids" (Jain and Dubes, 1988). With the centroid method, the new distances between

clusters can be calculated by the following Lance-Williams formula:

$$\begin{aligned} & D(C_k, C_i \cup C_j) \\ = & \frac{|C_i|}{|C_i| + |C_j|} D(C_k, C_i) + \frac{|C_j|}{|C_i| + |C_j|} D(C_k, C_j) \\ & - \frac{|C_i||C_j|}{(|C_i| + |C_j|)^2} D(C_i, C_j), \end{aligned} \tag{7.9}$$

where C_k, C_i, and C_j are three clusters in one level of clustering.

Let C and C' be any two nonoverlapping clusters, i.e., $C \cap C' = \phi$. Then it follows from equation (7.9) that

$$\begin{aligned} & D(C, C') \\ = & \frac{1}{|C||C'|} \sum_{\mathbf{x} \in C, \mathbf{y} \in C'} d(\mathbf{x}, \mathbf{y}) - \frac{1}{2|C|^2} \sum_{\mathbf{x}, \mathbf{y} \in C} d(\mathbf{x}, \mathbf{y}) \\ & - \frac{1}{2|C'|^2} \sum_{\mathbf{x}, \mathbf{y} \in C'} d(\mathbf{x}, \mathbf{y}), \end{aligned} \tag{7.10}$$

where $d(\cdot, \cdot)$ is the distance function by which the dissimilarity matrix is calculated.

In fact, let C_1, C_2, and C_3 be three nonempty, mutually nonoverlapping clusters, and assume

$$D(C_i, C_j) = \frac{1}{n_i n_j} \Sigma(C_i, C_j) - \frac{1}{2n_i^2} \Sigma(C_i) - \frac{1}{2n_j^2} \Sigma(C_j) \tag{7.11}$$

for $1 \le i < j \le 3$, where $n_i = |C_i|, n_j = |C_j|$, $\Sigma(C_i, C_j)$ is the total between-clusters distance of C_i and C_j, that is,

$$\Sigma(C_i, C_j) = \sum_{\mathbf{x} \in C_i, \mathbf{y} \in C_j} d(\mathbf{x}, \mathbf{y}),$$

$\Sigma(C_i)$ is the total within-cluster distance of C_i, that is,

$$\Sigma(C_i) = \sum_{\mathbf{x}, \mathbf{y} \in C_i} d(\mathbf{x}, \mathbf{y}),$$

and $\Sigma(C_j)$ is defined similarly.

We should prove that

$$\begin{aligned} & D(C_1, C_2 \cup C_3) \\ = & \frac{1}{n_1(n_2 + n_3)} \Sigma(C_1, C_2 \cup C_3) - \frac{1}{2n_1^2} \Sigma(C_1) \\ & - \frac{1}{2(n_2 + n_3)^2} \Sigma(C_2 \cup C_3). \end{aligned} \tag{7.12}$$

From equation (7.9), we have

$$\begin{aligned} & D(C_1, C_2 \cup C_3) \\ = & \frac{n_2}{n_2+n_3} D(C_1, C_2) + \frac{n_3}{n_2+n_3} D(C_1, C_3) \\ & - \frac{n_2 n_3}{(n_2+n_3)^2} D(C_2, C_3), \end{aligned}$$

Substituting equation (7.11) into the above equation and performing some simple manipulations, we have

$$\begin{aligned} & D(C_1, C_2 \cup C_3) \\ = & \frac{n_2}{n_2+n_3} \left(\frac{1}{n_1 n_2} \Sigma(C_1, C_2) - \frac{1}{2n_1^2} \Sigma(C_1) - \frac{1}{2n_2^2} \Sigma(C_2) \right) \\ & + \frac{n_3}{n_2+n_3} \left(\frac{1}{n_1 n_3} \Sigma(C_1, C_3) - \frac{1}{2n_1^2} \Sigma(C_1) - \frac{1}{2n_3^2} \Sigma(C_3) \right) \\ & - \frac{n_2 n_3}{(n_2+n_3)^2} \left(\frac{1}{n_2 n_3} \Sigma(C_2, C_3) - \frac{1}{2n_2^2} \Sigma(C_2) - \frac{1}{2n_3^2} \Sigma(C_3) \right) \\ = & \frac{1}{n_1(n_2+n_3)} \Sigma(C_1, C_2 \cup C_3) - \frac{1}{2n_1^2} \Sigma(C_1) \\ & - \frac{1}{2(n_2+n_3)^2} [\Sigma(C_2) + \Sigma(C_3) + 2\Sigma(C_2, C_3)] \\ = & \frac{1}{n_1(n_2+n_3)} \Sigma(C_1, C_2 \cup C_3) - \frac{1}{2n_1^2} \Sigma(C_1) - \frac{1}{2(n_2+n_3)^2} \Sigma(C_2 \cup C_3). \end{aligned}$$

In the above, we used

$$\Sigma(C_1, C_2) + \Sigma(C_1, C_3) = \Sigma(C_1, C_2 \cup C_3)$$

and

$$\Sigma(C_2) + \Sigma(C_3) + 2\Sigma(C_2, C_3) = \Sigma(C_2 \cup C_3).$$

In particular, if we take $d(\cdot, \cdot)$ in equation (7.10) as the squared Euclidean distance, then the distance $D(C, C')$ is exactly the squared Euclidean distance between the centroids of C and C'.

Actually, if $d(\cdot, \cdot)$ in equation (7.10) is the squared Euclidean distance, then we have

$$\begin{aligned} & D(C, C') \\ = & \frac{1}{|C||C'|} \sum_{\mathbf{x} \in C, \mathbf{y} \in C'} (\mathbf{x}-\mathbf{y})(\mathbf{x}-\mathbf{y})^T - \frac{1}{2|C|^2} \sum_{\mathbf{x}, \mathbf{y} \in C} (\mathbf{x}-\mathbf{y})(\mathbf{x}-\mathbf{y})^T \\ & - \frac{1}{2|C'|^2} \sum_{\mathbf{x}, \mathbf{y} \in C'} (\mathbf{x}-\mathbf{y})(\mathbf{x}-\mathbf{y})^T \end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{|C|}\sum_{\mathbf{x}\in C}\mathbf{x}\mathbf{x}^T - \frac{2}{|C||C'|}\sum_{\mathbf{x}\in C,\mathbf{y}\in C'}\mathbf{x}\mathbf{y}^T + \frac{1}{|C'|}\sum_{\mathbf{x}\in C'}\mathbf{x}\mathbf{x}^T \\
&\quad -\frac{1}{|C|}\sum_{\mathbf{x}\in C}\mathbf{x}\mathbf{x}^T + \frac{1}{|C|^2}\sum_{\mathbf{x},\mathbf{y}\in C}\mathbf{x}\mathbf{y}^T - \frac{1}{|C'|}\sum_{\mathbf{y}\in C'}\mathbf{y}\mathbf{y}^T \\
&\quad +\frac{1}{|C'|^2}\sum_{\mathbf{x},\mathbf{y}\in C'}\mathbf{x}\mathbf{y}^T \\
&= \frac{1}{|C|^2}\sum_{\mathbf{x},\mathbf{y}\in C}\mathbf{x}\mathbf{y}^T + \frac{1}{|C'|^2}\sum_{\mathbf{x},\mathbf{y}\in C'}\mathbf{x}\mathbf{y}^T - \frac{2}{|C||C'|}\sum_{\mathbf{x}\in C,\mathbf{y}\in C'}\mathbf{x}\mathbf{y}^T \\
&= \left(\frac{1}{|C|}\sum_{\mathbf{x}\in C}\mathbf{x} - \frac{1}{|C'|}\sum_{\mathbf{x}\in C'}\mathbf{x}\right)\left(\frac{1}{|C|}\sum_{\mathbf{y}\in C}\mathbf{y} - \frac{1}{|C'|}\sum_{\mathbf{y}\in C'}\mathbf{y}\right)^T,
\end{aligned}
$$

since $(\mathbf{x}-\mathbf{y})(\mathbf{x}-\mathbf{y})^T = \mathbf{x}\mathbf{x}^T - 2\mathbf{x}\mathbf{y}^T + \mathbf{y}\mathbf{y}^T$ and $\mathbf{x}\mathbf{y}^T = \mathbf{y}\mathbf{x}^T$.

Equation (7.10) provides another way to compute the distances between new clusters and old ones. Applying the centroid method to the data set given in Figure 7.9, the first stage is still the same as in other methods, i.e., $\mathbf{x}_1$ and $\mathbf{x}_2$ will be merged. After $\mathbf{x}_1$ and $\mathbf{x}_2$ are merged, the distances are updated as

$$
\begin{aligned}
D(\{\mathbf{x}_1,\mathbf{x}_2\},\mathbf{x}_3) &= \frac{1}{2}(d(\mathbf{x}_1,\mathbf{x}_3)+d(\mathbf{x}_2,\mathbf{x}_3)) - \frac{1}{4}d(\mathbf{x}_1,\mathbf{x}_2) = 2.245,\\
D(\{\mathbf{x}_1,\mathbf{x}_2\},\mathbf{x}_4) &= \frac{1}{2}(d(\mathbf{x}_1,\mathbf{x}_4)+d(\mathbf{x}_2,\mathbf{x}_4)) - \frac{1}{4}d(\mathbf{x}_1,\mathbf{x}_2) = 3.355,\\
D(\{\mathbf{x}_1,\mathbf{x}_2\},\mathbf{x}_5) &= \frac{1}{2}(d(\mathbf{x}_1,\mathbf{x}_5)+d(\mathbf{x}_2,\mathbf{x}_5)) - \frac{1}{4}d(\mathbf{x}_1,\mathbf{x}_2) = 2.895,
\end{aligned}
$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1,\mathbf{x}_2\}$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\{\mathbf{x}_1,\mathbf{x}_2\}$	0	2.245	3.355	2.895
$\mathbf{x}_3$	2.245	0	1.12	1.41
$\mathbf{x}_4$	3.355	1.12	0	1.5
$\mathbf{x}_5$	2.895	1.41	1.5	0

At the second stage, $\mathbf{x}_3$ and $\mathbf{x}_4$ will be merged, and the distances are updated as

$$
\begin{aligned}
D(\{\mathbf{x}_3,\mathbf{x}_4\},\{\mathbf{x}_1,\mathbf{x}_2\}) &= \frac{1}{2}(2.245+3.355) - \frac{1}{4}(1.12) = 2.52,\\
D(\{\mathbf{x}_3,\mathbf{x}_4\},\mathbf{x}_5) &= \frac{1}{2}(1.41+1.5) - \frac{1}{4}(1.12) = 1.175,
\end{aligned}
$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4\}$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.52	2.895
$\{\mathbf{x}_3, \mathbf{x}_4\}$	2.52	0	1.175
$\mathbf{x}_5$	2.895	1.175	0

At the third stage, $\mathbf{x}_5$ will be merged with $\{\mathbf{x}_3, \mathbf{x}_4\}$, and the distance is updated as

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) = \frac{2}{3}(2.52) + \frac{1}{3}(2.895) - \frac{2}{9}(1.175) = 2.384,$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.39
$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$	2.39	0

The distances can also be updated by equation (7.10). For example, the distance between $\{\mathbf{x}_1, \mathbf{x}_2\}$ and $\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$ in the last stage can be computed as

$$\begin{aligned}
&D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) \\
&= \frac{1}{6}(d_{13} + d_{14} + d_{15} + d_{23} + d_{24} + d_{25}) - \frac{1}{8}(2d_{12}) \\
&\quad - \frac{1}{18}(2d_{34} + 2d_{35} + 2d_{45}) \\
&= \frac{1}{6}(2.24 + 3.35 + 3 + 2.5 + 3.61 + 3.04) - \frac{1}{4}(0.5) \\
&\quad - \frac{1}{9}(1.12 + 1.41 + 1.5) \\
&= 2.957 - 0.125 - 0.448 \\
&= 2.384,
\end{aligned}$$

where $d_{ij} = d(\mathbf{x}_i, \mathbf{x}_j)$.

The whole process of this clustering can be represented by the dendrogram shown in Figure 7.14.

7.2.6 The Median Method

The median method is also referred to as the "weighted pair group method using centroids" (Jain and Dubes, 1988) or the "weighted centroid" method. It was first proposed by Gower (1967) in order to alleviate some disadvantages of the centroid method. In the centroid method, if the sizes of the two groups to be merged are quite different, then the centroid of the new group will be very close to that of the larger group and may remain within that group (Everitt, 1993). In the median method, the centroid of a new group is independent of the size of the groups that form the new group.

A disadvantage of the median method is that it is not suitable for measures such as correlation coefficients, since interpretation in a geometrical sense is no longer possible (Lance and Williams, 1967a).

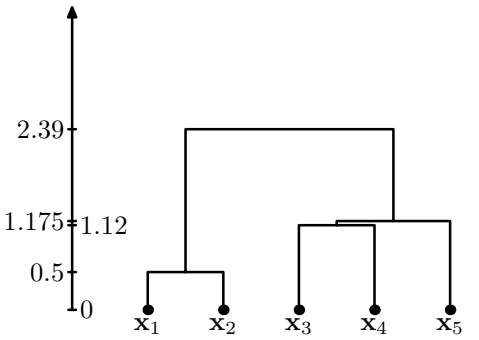

Figure 7.14. *The dendrogram produced by applying the centroid method to the data set given in Figure 7.9.*

In the median method, the distances between newly formed groups and other groups are computed as

$$D(C_k, C_i \cup C_j) = \frac{1}{2}D(C_k, C_i) + \frac{1}{2}D(C_k, C_j) - \frac{1}{4}D(C_i, C_j), \tag{7.13}$$

where C_k, C_i, and C_j are three clusters in one level of clustering.

We now take the data set given in Figure 7.9 as an example to illustrate the median method. The first stage of the median method is still the same as in other methods. After $\mathbf{x}_1$ and $\mathbf{x}_2$ are merged, the distances are updated as

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3) = \frac{1}{2}(d(\mathbf{x}_1, \mathbf{x}_3) + d(\mathbf{x}_2, \mathbf{x}_3)) - \frac{1}{4}d(\mathbf{x}_1, \mathbf{x}_2) = 2.245,$$
$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4) = \frac{1}{2}(d(\mathbf{x}_1, \mathbf{x}_4) + d(\mathbf{x}_2, \mathbf{x}_4)) - \frac{1}{4}d(\mathbf{x}_1, \mathbf{x}_2) = 3.355,$$
$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_5) = \frac{1}{2}(d(\mathbf{x}_1, \mathbf{x}_5) + d(\mathbf{x}_2, \mathbf{x}_5)) - \frac{1}{4}d(\mathbf{x}_1, \mathbf{x}_2) = 2.895,$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.245	3.355	2.895
$\mathbf{x}_3$	2.245	0	1.12	1.41
$\mathbf{x}_4$	3.355	1.12	0	1.5
$\mathbf{x}_5$	2.895	1.41	1.5	0

At the second stage of this method, $\mathbf{x}_3$ and $\mathbf{x}_4$ will be merged. After $\mathbf{x}_3$ and $\mathbf{x}_4$ are merged, the distances between clusters are updated as

$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \{\mathbf{x}_1, \mathbf{x}_2\}) = \frac{1}{2}(2.245 + 3.355) - \frac{1}{4}(1.12) = 2.52,$$
$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \mathbf{x}_5) = \frac{1}{2}(1.41 + 1.5) - \frac{1}{4}(1.12) = 1.175,$$

and the dissimilarity matrix becomes

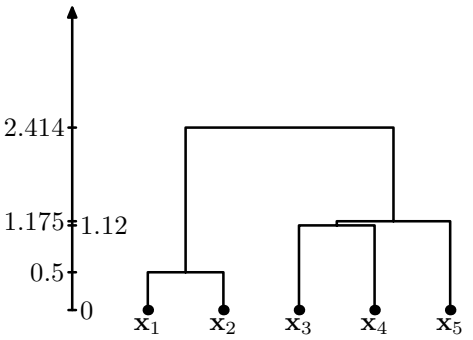

Figure 7.15. *The dendrogram produced by applying the median method to the data set given in Figure 7.9.*

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4\}$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.52	2.895
$\{\mathbf{x}_3, \mathbf{x}_4\}$	2.52	0	1.175
$\mathbf{x}_5$	2.895	1.175	0

Clusters $\{\mathbf{x}_3, \mathbf{x}_4\}$ and $\mathbf{x}_5$ will be merged at the third stage of this method. The distance is updated as

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) = \frac{1}{2}(2.52 + 2.895) - \frac{1}{4}(1.175) = 2.414,$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	2.414
$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$	2.414	0

The whole process of this clustering can be represented by the dendrogram shown in Figure 7.15.

7.2.7 Ward's Method

Ward Jr. (1963) and Ward Jr. and Hook (1963) proposed a hierarchical clustering procedure seeking to form the partitions $P_n, P_{n-1}, \ldots, P_1$ in a manner that minimizes the loss of information associated with each merging. Usually, the information loss is quantified in terms of an error sum of squares (ESS) criterion, so Ward's method is often referred to as the "minimum variance" method.

Given a group of data points C, the ESS associated with C is given by

$$\text{ESS}(C) = \sum_{\mathbf{x} \in C} (\mathbf{x} - \mu(C))(\mathbf{x} - \mu(C))^T,$$

or

$$\begin{aligned}\mathrm{ESS}(C) &= \sum_{\mathbf{x}\in C}\mathbf{x}\mathbf{x}^T - \frac{1}{|C|}\left(\sum_{\mathbf{x}\in C}\mathbf{x}\right)\left(\sum_{\mathbf{x}\in C}\mathbf{x}\right)^T \\ &= \sum_{\mathbf{x}\in C}\mathbf{x}\mathbf{x}^T - |C|\mu(C)\mu(C)^T, \end{aligned} \tag{7.14}$$

where $\mu(C)$ is the mean of C, that is,

$$\mu(C) = \frac{1}{|C|}\sum_{\mathbf{x}\in C}\mathbf{x}.$$

Suppose there are k groups $C_1, C_2, \ldots, C_k$ in one level of the clustering. Then the information loss is represented by the sum of ESSs given by

$$\mathrm{ESS} = \sum_{i=1}^{k}\mathrm{ESS}(C_i),$$

which is the total within-group ESS.

At each step of Ward's method, the union of every possible pair of groups is considered and two groups whose fusion results in the minimum increase in loss of information are merged.

If the squared Euclidean distance is used to compute the dissimilarity matrix, then the dissimilarity matrix can be updated by the Lance-Williams formula during the process of clustering as follows (Wishart, 1969):

$$\begin{aligned} &D(C_k, C_i \cup C_j) \\ &= \frac{|C_k| + |C_i|}{\Sigma_{ijk}}D(C_k, C_i) + \frac{|C_k| + |C_j|}{\Sigma_{ijk}}D(C_k, C_j) \\ &\quad -\frac{|C_k|}{\Sigma_{ijk}}D(C_i, C_j), \end{aligned} \tag{7.15}$$

where $\Sigma_{ijk} = |C_k| + |C_i| + |C_j|$.

To justify this, we suppose C_i and C_j are chosen to be merged and the resulting cluster is denoted by C_t, i.e., $C_t = C_i \cup C_j$. Then the increase in ESS is

$$\begin{aligned}\Delta\mathrm{ESS}_{ij} &= \mathrm{ESS}(C_t) - \mathrm{ESS}(C_i) - \mathrm{ESS}(C_j) \\ &= \left(\sum_{\mathbf{x}\in C_t}\mathbf{x}\mathbf{x}^T - |C_t|\mu_t\mu_t^T\right) - \left(\sum_{\mathbf{x}\in C_i}\mathbf{x}\mathbf{x}^T - |C_i|\mu_i\mu_i^T\right) \\ &\quad - \left(\sum_{\mathbf{x}\in C_t}\mathbf{x}\mathbf{x}^T - |C_j|\mu_j\mu_j^T\right) \\ &= |C_i|\mu_i\mu_i^T + |C_j|\mu_j\mu_j^T - |C_t|\mu_t\mu_t^T, \end{aligned} \tag{7.16}$$

where μ_t, μ_i, and μ_j are the means of clusters C_t, C_i, and C_j, respectively.

Noting that $|C_t|\mu_t = |C_i|\mu_i + |C_j|\mu_j$, squaring both sides of this equation gives

$$|C_t|^2\mu_t\mu_t^T = |C_i|^2\mu_i\mu_i^T + |C_j|^2\mu_j\mu_j^T + 2|C_i||C_j|\mu_i\mu_j^T,$$

or

$$\begin{aligned}|C_t|^2\mu_t\mu_t^T &= |C_i|^2\mu_i\mu_i^T + |C_j|^2\mu_j\mu_j^T + |C_i||C_j|(\mu_i\mu_i^T + \mu_j\mu_j^T)\\ &\quad -|C_i||C_j|(\mu_i-\mu_j)(\mu_i-\mu_j)^T\\ &= |C_i|(|C_i|+|C_j|)\mu_i\mu_i^T + |C_j|(|C_i|+|C_j|)\mu_j\mu_j^T\\ &\quad -|C_i||C_j|(\mu_i-\mu_j)(\mu_i-\mu_j)^T, \end{aligned} \tag{7.17}$$

since

$$2\mu_i\mu_j^T = \mu_i\mu_i^T + \mu_j\mu_j^T - (\mu_i-\mu_j)(\mu_i-\mu_j)^T.$$

Dividing both sides of equation (7.17) by $|C_t|$ and substituting $|C_t|\mu_t\mu_t^T$ into equation (7.16) give

$$\Delta\text{ESS}_{ij} = \frac{|C_i||C_j|}{|C_i|+|C_j|}(\mu_i-\mu_j)(\mu_i-\mu_j)^T. \tag{7.18}$$

Now considering the increase in ESS that would result from the potential fusion of groups C_k and C_t, from equation (7.18) we have

$$\Delta\text{ESS}_{kt} = \frac{|C_k||C_t|}{|C_k|+|C_j|}(\mu_k-\mu_t)(\mu_k-\mu_t)^T, \tag{7.19}$$

where $\mu_k = \mu(C_k)$ is the mean of group C_k.

Noting that $\mu_t = \frac{1}{|C_t|}(|C_i|\mu_i + |C_j|\mu_j)$ and $|C_t| = |C_i| + |C_j|$, using equation (7.17), we have

$$\begin{aligned}&(\mu_k-\mu_t)(\mu_k-\mu_t)^T\\ &= \frac{|C_i|}{|C_t|}(\mu_k-\mu_i)(\mu_k-\mu_i)^T + \frac{|C_j|}{|C_t|}(\mu_k-\mu_j)(\mu_k-\mu_j)^T\\ &\quad -\frac{|C_i||C_j|}{|C_t|^2}(\mu_i-\mu_j)(\mu_i-\mu_j)^T.\end{aligned}$$

Substituting the above equation into equation (7.19), and after simple manipulations, we get

$$\begin{aligned}\Delta\text{ESS}_{kt} &= \frac{|C_k||C_i|}{|C_k|+|C_t|}(\mu_k-\mu_i)(\mu_k-\mu_i)^T\\ &\quad +\frac{|C_k||C_j|}{|C_k|+|C_t|}(\mu_k-\mu_j)(\mu_k-\mu_j)^T\\ &\quad -\frac{|C_k||C_i||C_j|}{|C_k|+|C_t|}(\mu_i-\mu_j)(\mu_i-\mu_j)^T,\end{aligned}$$

and, using equation (7.18), we have

$$\begin{aligned}\Delta\text{ESS}_{kt} &= \frac{|C_k|+|C_i|}{|C_k|+|C_t|}\Delta\text{ESS}_{ki} + \frac{|C_k|+|C_j|}{|C_k|+|C_t|}\Delta\text{ESS}_{kj}\\ &\quad -\frac{|C_k|}{|C_k|+|C_t|}\Delta\text{ESS}_{ij}.\end{aligned} \tag{7.20}$$

Table 7.6. *The dissimilarity matrix of the data set given in Figure 7.9, where the entry (i, j) in the matrix is the squared Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$.*

	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\mathbf{x}_1$	0	0.25	5	11.25	9
$\mathbf{x}_2$	0.25	0	6.25	13	9.25
$\mathbf{x}_3$	5	6.25	0	1.25	2
$\mathbf{x}_4$	11.25	13	1.25	0	2.25
$\mathbf{x}_5$	9	9.25	2	2.25	0

This proves equation (7.15). If we compute the dissimilarity matrix for a data set $D = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ using the squared Euclidean distance, then the entry (i, j) of the dissimilarity matrix is

$$d_{ij}^2 = d(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T = \sum_{l=1}^{d} (x_{il} - x_{jl})^2,$$

where d is the dimensionality of the data set D.

If $C_i = \{\mathbf{x}_i\}$ and $C_j = \{\mathbf{x}_j\}$ in equation (7.18), then the increase in ESS that results from the fusion of $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$\Delta\text{ESS}_{ij} = \frac{1}{2} d_{ij}^2.$$

Since the objective of Ward's method is to find at each stage those two groups whose fusion gives the minimum increase in the total within-group ESS, the two points with minimum squared Euclidean distance will be merged at the first stage. Suppose $\mathbf{x}_i$ and $\mathbf{x}_j$ have minimum squared Euclidean distance. Then $C_i = \{\mathbf{x}_i\}$ and $C_j = \{\mathbf{x}_j\}$ will be merged. After C_i and C_j are merged, the distances between $C_i \cup C_j$ and other points must be updated.

Now let $C_k = \{\mathbf{x}_k\}$ be any other group. Then the increase in ESS that would result from the potential fusion of C_k and $C_i \cup C_j$ can be calculated from equation (7.20) as

$$\Delta\text{ESS}_{k(ij)} = \frac{2}{3}\frac{d_{ki}^2}{2} + \frac{2}{3}\frac{d_{kj}^2}{2} - \frac{1}{3}\frac{d_{ij}^2}{2}.$$

If we update the dissimilarity matrix using equation (7.15), then we have from the above equation that

$$\Delta\text{ESS}_{k(ij)} = \frac{1}{2} D(C_k, C_i \cup C_j).$$

Thus if we update the dissimilarity matrix using equation (7.15) during the process of clustering, then the two groups with minimum distance will be merged.

Taking the data set given in Figure 7.9 as an example, the dissimilarity matrix computed by the squared Euclidean distance is given in Table 7.6.

Initially, each single point forms a cluster and the total ESS is $\text{ESS}_0 = 0$. According to the above discussion, $\mathbf{x}_1$ and $\mathbf{x}_2$ will be merged at the first stage of Ward's method, and

the increase in ESS that results from the fusion of $\mathbf{x}_1$ and $\mathbf{x}_2$ is $\Delta\text{ESS}_{12} = \frac{1}{2}(0.25) = 0.125$; hence, the ESS becomes

$$\text{ESS}_1 = \text{ESS}_0 + \Delta\text{ESS}_{12} = 0.125.$$

Using equation (7.15), the distances are updated as

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_3) = \frac{2}{3}(d(\mathbf{x}_1, \mathbf{x}_3) + d(\mathbf{x}_2, \mathbf{x}_3)) - \frac{1}{3}d(\mathbf{x}_1, \mathbf{x}_2) = 7.42,$$
$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_4) = \frac{2}{3}(d(\mathbf{x}_1, \mathbf{x}_4) + d(\mathbf{x}_2, \mathbf{x}_4)) - \frac{1}{3}d(\mathbf{x}_1, \mathbf{x}_2) = 16.08,$$
$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{x}_5) = \frac{2}{3}(d(\mathbf{x}_1, \mathbf{x}_5) + d(\mathbf{x}_2, \mathbf{x}_5)) - \frac{1}{3}d(\mathbf{x}_1, \mathbf{x}_2) = 12.08,$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	7.42	16.08	12.08
$\mathbf{x}_3$	7.42	0	1.25	2
$\mathbf{x}_4$	16.08	1.25	0	2.25
$\mathbf{x}_5$	12.08	2	2.25	0

At the second stage of this method, $\mathbf{x}_3$ and $\mathbf{x}_4$ will be merged and the resulting increase in ESS is $\Delta\text{ESS}_{34} = \frac{1}{2}(1.25) = 0.625$. The total ESS becomes

$$\text{ESS}_2 = \text{ESS}_1 + \Delta\text{ESS}_{34} = 0.125 + 0.625 = 0.75.$$

After $\mathbf{x}_3$ and $\mathbf{x}_4$ are merged, the distances are updated as

$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \{\mathbf{x}_1, \mathbf{x}_2\}) = \frac{3}{4}(7.42 + 16.08) - \frac{2}{4}(1.25) = 17,$$
$$D(\{\mathbf{x}_3, \mathbf{x}_4\}, \mathbf{x}_5) = \frac{2}{3}(2 + 2.25) - \frac{1}{3}(1.25) = 2.42,$$

and the dissimilarity matrix becomes

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4\}$	$\mathbf{x}_5$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	17	12.08
$\{\mathbf{x}_3, \mathbf{x}_4\}$	17	0	2.42
$\mathbf{x}_5$	12.08	2.42	0

At the third stage, $\{\mathbf{x}_3, \mathbf{x}_4\}$ and $\mathbf{x}_5$ will be merged. The resulting increase in ESS is $\Delta\text{ESS}_{(34)5} = \frac{1}{2}(2.42) = 1.21$. Then the total ESS becomes

$$\text{ESS}_3 = \text{ESS}_2 + \Delta\text{ESS}_{(34)5} = 0.75 + 1.21 = 1.96.$$

The distances are updated as

$$D(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) = \frac{4}{5}(17) + \frac{3}{5}(12.08) - \frac{2}{5}(2.42) = 19.88,$$

and the dissimilarity matrix becomes

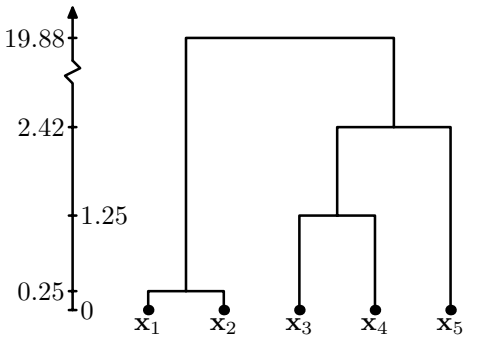

Figure 7.16. *The dendrogram produced by applying Ward's method to the data set given in Figure 7.9.*

	$\{\mathbf{x}_1, \mathbf{x}_2\}$	$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$
$\{\mathbf{x}_1, \mathbf{x}_2\}$	0	19.88
$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$	19.88	0

When all the data points are merged to form a single cluster, the increase in ESS will be $\Delta\text{ESS}_{(12)(345)} = \frac{1}{2}(19.88) = 9.94$ and the total ESS will be

$$\text{ESS}_4 = \text{ESS}_3 + \Delta\text{ESS}_{(12)(345)} = 1.96 + 9.94 = 11.9.$$

The whole process of this clustering can be represented by the dendrogram shown in Figure 7.16.

As pointed out by Anderberg (1973), a set of k clusters produced by Ward's method may or may not give the minimum possible ESS over all possible sets of k clusters formed from the n objects. However, the results of Ward's method are usually very good approximations of the optimal one. Kuiper and Fisher (1975) presented a comparison of Ward's method with the other five hierarchical clustering algorithms using the Monte Carlo method.

7.2.8 Other Agglomerative Methods

The seven agglomerative methods discussed in the previous subsections are most widely used in practice. In the Lance-Williams framework, there are some other agglomerative methods, such as the flexible method (Lance and Williams, 1967a), the sum of squares method (Jambu, 1978), and the mean dissimilarity method (Podani, 1989). Relevant discussions can be found in (Holman, 1992) and (Gordon, 1996).

There also exist agglomerative methods that cannot fit into the Lance-Williams framework. An example of such agglomerative methods is the bicriterion analysis proposed by Delattre and Hansen (1980).

7.3 Divisive Hierarchical Methods

The divisive hierarchical method proceeds the opposite way of the agglomerative hierarchical method. Initially, all the data points belong to a single cluster. The number of clusters

is increased by one at each stage of the algorithm by dividing an existing cluster into two clusters according to some criteria. A divisive hierarchical method may be adopted in which a single cluster is subdivided into smaller and smaller clusters. Divisive hierarchical clustering methods are essentially of two types: monothetic and polythetic (Everitt, 1993; Willett, 1988). A monothetic method divides the data on the basis of the possession or otherwise of a single specified attribute, while a polythetic method divides the data based on the values taken by all attributes.

It is not feasible to enumerate all possible divisions of a large (even moderate) cluster C to find the optimal partition, since there are $2^{|C|-1} - 1$ nontrivial different ways of dividing the cluster C into two clusters (Edwards and Cavalli-Sforza, 1965). Scott and Symons (1971a) have proposed an improved algorithm that requires examination of $2^d - 2$ partitions by assigning points in the hyperplane to the two clusters being considered, where d is the dimensionality of the data. Except for low-dimensional data, the algorithm proposed by Scott and Symons (1971a) is also very time-consuming. In fact, it turns out that the problem of finding an optimal bipartition for some clustering criteria is NP-hard (Gordon, 1996).

Another problem with divisive hierarchical algorithms is monotonicity, to be specified below. In a divisive hierarchy, one cluster is divided at a time, so what is the next cluster to be divided? This depends on the definition of a level. Such a level must be meaningful and monotone, which means that no subcluster may have a larger level than the level of its parent cluster.

However, it is possible to construct divisive hierarchical algorithms that do not need to consider all divisions and are monothetic. An algorithm called DIANA (DIvisive ANAlysis) presented in (Kaufman and Rousseeuw, 1990) is a divisive hierarchical clustering algorithm. It was based on the idea of Macnaughton-Smith et al. (1964). Other divisive hierarchical techniques are presented in (Edwards and Cavalli-Sforza, 1965) and (Späth, 1980).

7.4 Several Hierarchical Algorithms

Although the computational complexity of hierarchical algorithms is generally higher than that of partitional algorithms, many hierarchical algorithms have been designed and studied in (Kaufman and Rousseeuw, 1990), (Murtagh, 1983), and (Willett, 1988). In this section, we shall introduce some hierarchical algorithms, but we defer the introduction of hierarchical algorithms described in terms of graph theory (i.e., graph-based algorithms) to later chapters.

7.4.1 SLINK

The SLINK algorithm (Sibson, 1973) is a single-link hierarchical algorithm that can be carried out using arbitrary dissimilarity coefficients. In this algorithm, a compact representation of a dendrogram called a pointer representation, which offers economy in computation, is introduced.

Recall that a dendrogram is a nested sequence of partitions with associated numerical levels that can be defined as a function $c : [0, \infty) \to E(D)$ that satisfies

$$c(h) \subseteq c(h') \text{ if } h \leq h',$$
$$c(h) \text{ is eventually in } D \times D,$$
$$c(h + \delta) = c(h) \text{ for some small } \delta > 0,$$

where D is a given data set and $E(D)$ is the set of equivalence relations on D.

Recall also that a pointer representation is the pair of functions $\pi : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ and $\lambda : \pi(\{1, 2, \ldots, n\}) \to [0, \infty]$ that have the following properties:

$$\pi(n) = n, \quad \pi(i) > i \text{ for } i < n, \tag{7.21a}$$

$$\lambda(n) = \infty, \quad \lambda(\pi(i)) > \lambda(i) \text{ for } i < n, \tag{7.21b}$$

where n is the number of data points in D.

As discussed before, there is a one-to-one correspondence between dendrograms and pointer representations (Sibson, 1973). In particular, if c is a dendrogram, then the corresponding pointer representation is defined by

$$\lambda(i) = \inf\{h : \exists j > i \text{ such that } (i, j) \in c(h)\},$$
$$\pi(i) = \max\{j : (i, j) \in c(\lambda(i))\}$$

for $i < n$. Intuitively, $\lambda(i)$ is the lowest level at which the ith object is no longer the last object in its cluster, and $\pi(i)$ is the last object in the cluster that it joins. The pointer representation of a dendrogram ensures that a new object can be inserted in an efficient way.

ALGORITHM 7.1. The SLINK algorithm.

Require: n: number of objects; $d(i, j)$: dissimilarity coefficients;
1: $\Pi[1] \Leftarrow 1, \Lambda[1] \Leftarrow \infty$;
2: **for** $t = 1$ to $n - 1$ **do**
3: $\Pi[t + 1] \Leftarrow t + 1, \Lambda[t + 1] \Leftarrow \infty$;
4: $M[i] \Leftarrow d(i, t + 1)$ for $i = 1, 2, \ldots, t$;
5: **for** $i = 1$ to t **do**
6: **if** $\Lambda[i] \geq M[i]$ **then**
7: $M[\Pi[i]] \Leftarrow \min\{M[\Pi[i]], \Lambda[i]\}$;
8: $\Pi[i] \Leftarrow M[i], \Pi[i] \Leftarrow t + 1$;
9: **else**
10: $M[\Pi[i]] \Leftarrow \min\{M[\Pi[i]], M[i]\}$;
11: **end if**
12: **end for**
13: **for** $i = 1$ to t **do**
14: **if** $\Lambda[i] \geq \Lambda[\Pi[i]]$ **then**
15: $\Pi[i] \leq t + 1$;
16: **end if**
17: **end for**

18: **end for**

Let the dendrogram for the first t objects in the data set be c_t and its pointer representation be π_t, λ_t. Let $\mu_t(i)$ be defined recursively on i as

$$\mu_t(i) = \min\{d(i, t+1), \min_{j,\pi_t(j)=i} \max\{\mu_t(j), \lambda_t(j)\}\},$$

which is defined for all $i = 1, 2, \ldots, t$. It can be shown that the pointer representation for c_{t+1} is defined as (Sibson, 1973)

$$\pi_{t+1}(i) = \begin{cases} t+1 & \text{if } i = t+1, \\ t+1 & \text{if } \mu_t(i) \text{ or } \mu_t(\pi_t(i)) \leq \lambda_t(i) \text{ for } i < t+1, \\ \pi_t(i) & \text{otherwise,} \end{cases} \tag{7.22a}$$

$$\lambda_{t+1}(i) = \begin{cases} \min\{\mu_t(i), \lambda_t(i)\} & \text{if } i < t+1, \\ \infty & \text{if } i = t+1. \end{cases} \tag{7.22b}$$

If we start with $t = 1$ by $\pi_1(1) = 1, \lambda_1(1) = \infty$, then after $n-1$ steps of the recursive process defined in (7.22), we shall obtain π_n, λ_n, which is the pointer representation of the dendrogram on the whole data set. Let Π, Λ, and M be three n-dimensional vectors such that Π, Λ contain π_t, λ_t in their first t entries in the tth step. Then the SLINK algorithm can be described as in Algorithm 7.1. The number of operations to find π_n, λ_n is $O(n^2)$.

7.4.2 Single-link Algorithms Based on Minimum Spanning Trees

A single-link cluster analysis can be carried out by using only the information contained in the minimum spanning tree (MST) (Gower and Ross, 1969). The performance of single-link cluster analysis can be improved by incorporating the MST algorithms into the clustering. To present the details, we start with a brief introduction of MSTs and some efficient algorithms for finding an MST.

The tree is a concept in graph theory. A tree is a connected graph with no cycles (Jain and Dubes, 1988). A spanning tree is a tree containing all vertices of the graph. When each edge in a graph is weighted by the dissimilarity between the two vertices that the edge connects, the weight of a tree is the sum of the edge weights in the tree. An MST of a graph G is a tree that has minimal weight among all other spanning trees of G.

A number of algorithms have been developed to find an MST. Most popular algorithms to find the MST proceed iteratively. Two popular algorithms for finding an MST have been discussed in (Gower and Ross, 1969). These algorithms are also presented in (Kruskal, 1956) and (Prim, 1957). In these algorithms, the edges belong to one of two sets A and B at any stage, where A is the set containing the edges assigned to the MST and B is the set of edges not assigned. Prim (1957) suggested an iterative algorithm that starts with any one of the given vertices and initially assigns to A the shortest segment starting from this vertex. Then the algorithm continues to assign to A the shortest segment from B that connects at least one segment from A without forming a closed loop among the segments already in A. The algorithm will stop when there are $n-1$ segments in A. The MST produced by these algorithms may not be unique if there exist equal segments of minimum length.

Single-link cluster analysis can be performed by using a dissimilarity matrix that contains $\frac{n(n-1)}{2}$ distances, but it is impractical to record all $\frac{n(n-1)}{2}$ distances when n is large, where n is the number of data points. As such, the MST constructed from this data set provides a useful ancillary technique. To employ this technique in single-link cluster analysis, an MST should be constructed from the data set first.

Gower and Ross (1969) presented an approach to computing the MST from a data set. Let $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a data set, and let L_1, L_2, and L_3 be three n-dimensional vectors defined as follows:

$$L_1(i) = j \text{ if } \mathbf{x}_i \text{ is the } j\text{th point to join } A,$$
$$L_2(i) = j \text{ if } \mathbf{x}_i \text{ was linked to } \mathbf{x}_j \in A \text{ when it joined } A,$$
$$L_3(i) = d(\mathbf{x}_i, \mathbf{x}_{L_2(i)})$$

for $i = 1, 2, \ldots, n$, where $d(\cdot, \cdot)$ is a distance function.

The three vectors can be computed as follows. Initially, $L_1(1) = 1$, $L_1(i) = 0$ for $i = 2, 3, \ldots, n$, and L_2 and L_3 are set to the zero vector, i.e., $A = \{\mathbf{x}_1\}$. At the first stage, let i_1 be defined as

$$i_1 = \arg \max_{1 \le i \le n} L_1(i)$$

and let j_1 be defined as

$$j_1 = \arg \min_{1 \le j \le n, L_1(j)=0} d(\mathbf{x}_{i_1}, \mathbf{x}_j).$$

Then $\mathbf{x}_{j_1}$ will be assigned to A; i.e., $L_1(2) = 0$ will be changed to $L_1(j_1) = 2$, and L_2 and L_3 will be updated as $L_2(j_1) = i_1$, and $L_3(j_1) = d(\mathbf{x}_{j_1}, \mathbf{x}_{i_1})$.

At the rth stage, let i_r and j_r be computed as

$$i_r = \arg \max_{1 \le i \le n} L_1(i),$$
$$j_r = \arg \min_{1 \le j \le n, L_1(j)=0} d(\mathbf{x}_{i_r}, \mathbf{x}_j).$$

Hence, i_r is the point added to A at the previous stage. At this stage, $\mathbf{x}_{j_r}$ will be added to A, and the vectors will be updated as $L_1(j_r) = r + 1$, $L_2(j_r) = i_r$, and $L_3(j_r) = d(\mathbf{x}_{j_r}, \mathbf{x}_{i_r})$.

After $n - 1$ stages, all the points will be added to A; then the MST is found. Once the MST of the data set is found, the single-link cluster analysis can be performed from the MST. The algorithm does not need the full distance matrix at every level of clustering.

To find clusters at the first level, let δ be a distance threshold and L_0 be the largest multiple of δ that is less than the minimum distance, and let H be a set of links whose lengths lie between L_0 and $L_0 + \delta$. Let G be a list that contains the group members contiguously, marking the final member of each group with an indicator. For each link in H, find the endpoints in G, agglomerate the two groups in which the points are found, and shift down the intervening groups when necessary. Using the same procedure, a hierarchical system of agglomerative clusters can be easily constructed. This algorithm is also presented in (Rohlf, 1973), along with the FORTRAN code.

7.4.3 CLINK

Like the SLINK algorithm (Sibson, 1973), the CLINK algorithm (Defays, 1977) is also a hierarchical clustering algorithm based on a compact representation of a dendrogram.

But the CLINK algorithm is designed for the complete link method. The input of the CLINK algorithm is a fuzzy relation R, and the output is the pointer representation of the dendrogram.

A fuzzy relation R used in CLINK is defined as a collection of ordered pairs. For example, given a data set $D = \{\mathbf{x}_i : i = 1, 2, \ldots, n\}$, the fuzzy relation R on D can be characterized as a membership function $R(\cdot, \cdot)$ that associates with each pair (i, j) the dissimilarity measure from $\mathbf{x}_i$ to $\mathbf{x}_j$. The fuzzy relations R used in CLINK are reflexive and symmetric, i.e.,

$$R(i, i) = 0, \quad 1 \le i \le n \text{ (reflexivity)},$$
$$R(i, j) = R(j, i), \quad 1 \le i, j \le n \text{ (symmetry)}.$$

Let R and Q be two fuzzy relations defined on D. Then the min-max composition of R and Q is defined by

$$R \circ Q(i, j) = \min\{\max\{Q(i, k), R(k, j)\} : k = 1, 2, \ldots, n\}$$

for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$.

The r-fold composition $R \circ R \circ \cdots \circ R$ is denoted by R^r. R is said to be transitive if $R^2 \supset R$. An ultrametric relation is a fuzzy relation that is reflexive, symmetric, and transitive. If a fuzzy relation R is reflexive and symmetric, then its transitive closure $\bar{R} = R^{n-1}$ may be obtained by a single-linkage clustering. A complete linkage clustering gives one or more minimal ultrametric relations (MURs) superior to R.

The goal of the CLINK algorithm is to find one of the MURs L superior to a reflexive symmetric fuzzy relation R. L and $\bar{R}$ can be viewed as two extreme clusterings of D. In order to find such an L efficiently, the pointer representation (Sibson, 1973) is used in the CLINK algorithm. The pointer representation is a pair of functions (π, λ) defined as in equation (7.21). There is a one-to-one correspondence between pointer representations and ultrametric relations.

ALGORITHM 7.2. The pseudocode of the CLINK algorithm.

Require: n: number of objects; $R(i, j)$: dissimilarity coefficients;
1: $\Pi[1] \Leftarrow 1, \Lambda[1] \Leftarrow \infty$;
2: **for** $t = 1$ to $n - 1$ **do**
3: $\quad \Pi[t + 1] \Leftarrow t + 1, \Lambda[t + 1] \Leftarrow \infty$;
4: $\quad M[i] \Leftarrow R(i, t + 1)$ for $i = 1, 2, \ldots, t$;
5: $\quad$ **for** $i = 1$ to t **do**
6: $\quad\quad$ **if** $\Lambda[i] < M[i]$ **then**
7: $\quad\quad\quad M[\Pi[i]] \Leftarrow \max\{M[\Pi[i]], M[i]\}$;
8: $\quad\quad\quad M[i] \Leftarrow \infty$;
9: $\quad\quad$ **end if**
10: $\quad$ **end for**
11: $\quad$ Set $a \Leftarrow t$;
12: $\quad$ **for** $i = 1$ to t **do**
13: $\quad\quad$ **if** $\Lambda[t - i + 1] \ge \Lambda[\Pi[t - i + 1]]$ **then**

```
14:          if M[t − i + 1] < M[a] then
15:            a ⇐ t − i + 1;
16:          end if
17:        else
18:          M[t − i + 1] ⇐ ∞;
19:        end if
20:      end for
21:      Set b ⇐ Π[a], c ⇐ Λ[a], Π[a] ⇐ t + 1 and Λ[a] ⇐ M[a];
22:      if a < t then
23:        while b < t do
24:          Set d ⇐ Π[b], e ⇐ Λ[b], Π[b] ⇐ t + 1 and Λ[b] ⇐ c;
25:          Set b ⇐ d and c ⇐ e;
26:        end while
27:        if b = t then
28:          Set Π[b] ⇐ t + 1 and Λ[b] ⇐ c;
29:        end if
30:      end if
31:      for i = 1 to t do
32:        if Π[Π[i]] = t + 1 and Λ[i] = Λ[Π[i]] then
33:          Set Π[i] ⇐ t + 1;
34:        end if
35:      end for
36:    end for
```

Suppose that L is an ultrametric relation on D. Then the corresponding pointer representation (π, λ) is defined as

$$\pi(i) = \begin{cases} \max\{j : L(i, j) = \lambda(i)\} & \text{if } i < n, \\ n & \text{if } i = n, \end{cases}$$

$$\lambda(i) = \begin{cases} \min\{L(i, j) : j > i\} & \text{if } i < n, \\ \infty & \text{if } i = n. \end{cases}$$

Conversely, if (π, λ) is a pointer representation, then the corresponding ultrametric relation R is defined as

$$R(i, j) = \begin{cases} \lambda(i) & \text{if } j = \pi(i) > i, \\ \lambda(j) & \text{if } i = \pi(j) > j, \\ 0 & \text{if } i = j, \\ \infty & \text{otherwise.} \end{cases}$$

Let R_t be the restriction of R to the first t data points of D. If L_t is a MUR superior to R_t, then a MUR superior to R_{t+1} can be easily obtained from L_t. To do this, let (π_t, λ_t) be the pointer representation of a MUR L_t superior to R_t. The pseudocode of the CLINK algorithm is described in Algorithm 7.2. It is modified from the SLINK algorithm. These two algorithms cannot be further improved, because they require all pairwise dissimilarities to be considered (Murtagh, 1983).

7.4.4 BIRCH

Zhang et al. (1996) proposed an agglomerative hierarchical algorithm, called BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies), for clustering very large numerical data sets in Euclidean spaces. It is also the first clustering algorithm in the database area that takes account of noise.

In the algorithm of BIRCH, a clustering feature (CF) vector is used to summarize the information of each cluster. Given a cluster C of a d-dimensional data set, the CF vector for C is a triple defined as

$$\mathrm{CF}(C) = (|C|, S_1, S_2),$$

where $|C|$ is the number of instances in C, and S_1 and S_2 are d-dimensional vectors defined as

$$S_1 = \sum_{\mathbf{x}\in C} \mathbf{x} = \left(\sum_{\mathbf{x}\in C} x_1, \sum_{\mathbf{x}\in C} x_2, \ldots, \sum_{\mathbf{x}\in C} x_d \right),$$
$$S_2 = \sum_{\mathbf{x}\in C} \mathbf{x}^2 = \left(\sum_{\mathbf{x}\in C} x_1^2, \sum_{\mathbf{x}\in C} x_2^2, \ldots, \sum_{\mathbf{x}\in C} x_d^2 \right),$$

where x_j $(1 \le j \le d)$ is the value of the jth attribute of $\mathbf{x}$.

At the beginning, a CF tree is built dynamically as new data objects are inserted. A CF tree has three parameters: the branching factor B, the leaf factor L, and the threshold T. Each nonleaf node contains at most B subnodes of the form $[\mathrm{CF}_i, child_i]$, each leaf node contains at most L entries of the form $[\mathrm{CF}_i]$, and the diameter of each entry in a leaf node has to be less than T.

Outliers or noise are determined by considering the density of each entry in leaf nodes; i.e., low-density entries of leaf nodes are treated as outliers. Potential outliers are written out to disk in order to reduce the size of the tree. At certain points in the process, outliers are scanned to see if they can be reabsorbed into the current tree without causing the tree to grow in size.

After the CF tree is built, an agglomerative hierarchical clustering algorithm is applied directly to the nodes represented by their CF vectors. Then for each cluster, a centroid is obtained. Finally, a set of new clusters is formed by redistributing each data point to its nearest centroid.

BIRCH works well when clusters are of convex or spherical shape and uniform size. However, it is unsuitable when clusters have different sizes or nonspherical shapes (Guha et al., 1998).

7.4.5 CURE

Guha et al. (1998) proposed an agglomerative hierarchical clustering algorithm called CURE (Clustering Using REpresentatives) that can identify nonspherical shapes in large databases and wide variance in size. In this algorithm, each cluster is represented by a certain fixed number of points that are well scattered in the cluster. A combination of random sampling and partitioning is used in CURE in order to handle large databases.

CURE consists of six main steps: draw a random sample, partition the sample, partially cluster the partitions, eliminate the outliers, cluster the partial clusters, and label the data on the disk. Since CURE was developed for large databases, it begins by drawing a random sample from the database. Then the sample is partitioned and data points in each partition are partially clustered. After the outliers are eliminated, the preclustered data in each partition are clustered in a final pass to generate the final clusters.

In the first step, a random sample is drawn in order to handle large databases. In this step, the Chernoff bounds (Motwani and Raghavan, 1995) are used to analytically derive values for sample sizes such that the probability of missing clusters is low. The following theorem is proved.

Theorem 7.1. *For a cluster C, if the sample size s satisfies*

$$s \geq fn + \frac{n}{|C|}\log\left(\frac{1}{\delta}\right) + \frac{n}{|C|}\sqrt{\left(\log\frac{1}{\delta}\right)^2 + 2f|C|\log\frac{1}{\delta}},$$

then the probability that the sample contains fewer than $f|C|$ $(0 \leq f \leq 1)$ points belonging to C is less than δ, $0 \leq \delta \leq 1$.

In the second step, a simple partitioning scheme is proposed for speeding up CURE when input sizes become large, since samples of larger sizes are required in some situations, such as when separation between clusters decreases and clusters become less densely packed. This is done by partitioning the sample space into p partitions, each of size $\frac{s}{p}$, where s is the sample size.

In the third step, each partition is clustered until the final number of clusters in each partition reduces to $\frac{s}{pq}$ for some constant $q > 1$. Since data sets almost always contain outliers, the fourth step of CURE is to eliminate outliers. The idea of this step is based on the fact that outliers tend, due to their large distances from other points, to merge with other points less and typically grow at a much slower rate than actual clusters in an agglomerative hierarchical clustering.

$\frac{s}{pq}$ clusters are generated for each partition in the third step. In the fifth step, a second clustering pass is run on the $\frac{s}{q}$ partial clusters for all the partitions. The hierarchical clustering algorithm is used only on points in a partition.

Since the input to CURE's clustering algorithm is a set of randomly sampled points from the original data set, the last step is to assign the appropriate cluster labels to the remaining data points such that each data point is assigned to the cluster containing the representative point closest to it.

The space complexity of CURE is linear in the input size n. The worst-case time complexity is $O(n^2 \log n)$, which can be further reduced to $O(n^2)$ in lower-dimensional spaces (e.g. two-dimensional space).

7.4.6 DIANA

DIANA (DIvisive ANAlysis) presented in (Kaufman and Rousseeuw, 1990, Chapter 6) is a divisive hierarchical algorithm based on the proposal of Macnaughton-Smith et al.

(1964). It can be applied to all data sets that can be clustered by means of the agglomerative hierarchical algorithms.

The algorithm DIANA proceeds by a series of successive splits. At each step, the biggest cluster (i.e., the cluster with the largest diameter, which is defined to be the largest dissimilarity between two objects in it) is divided until at step $n-1$ each object is in a single cluster.

Let C be a cluster. Then the diameter of C is defined to be

$$\text{Diam}(C) = \max_{\mathbf{x},\mathbf{y}\in C} d(\mathbf{x}, \mathbf{y}). \tag{7.23}$$

The values of the diameter are also used as heights to represent the clustering structure in the dendrograms or banners.

At each step, let C ($|C| \geq 2$) be the cluster to be divided, and let A and B be the clusters divided from C, i.e., $A \cap B = \Phi$ and $A \cup B = C$. Initially, $A = C$ and $B = \Phi$, and the algorithm DIANA finds A and B by moving points from A to B iteratively. At the first stage, a point $\mathbf{y}_1$ will be moved from A to B if it maximizes the function

$$D(\mathbf{x}, A\backslash\{\mathbf{x}\}) = \frac{1}{|A|-1} \sum_{\mathbf{y}\in A, \mathbf{y}\neq\mathbf{x}} d(\mathbf{x}, \mathbf{y}), \tag{7.24}$$

where $d(\cdot, \cdot)$ can be any distance measure appropriate for the data.

Then A and B are updated as

$$A_{new} = A_{old}\backslash\{\mathbf{y}_1\},$$
$$B_{new} = B_{old} \cup \{\mathbf{y}_1\}.$$

In the next stage, the algorithm looks for other points in A that should be moved to B. Let $\mathbf{x} \in A$, and let the test function be defined as

$$\begin{aligned} &D(\mathbf{x}, A\backslash\{\mathbf{x}\}) - D(\mathbf{x}, B) \\ &= \frac{1}{|A|-1} \sum_{\mathbf{y}\in A, \mathbf{y}\neq\mathbf{x}} d(\mathbf{x}, \mathbf{y}) - \frac{1}{|B|} \sum_{\mathbf{z}\in B} d(\mathbf{x}, \mathbf{z}). \end{aligned} \tag{7.25}$$

If a point $\mathbf{y}_2$ maximizes the function in equation (7.25) and the maximal value is strictly positive, then $\mathbf{y}_2$ will be moved from A to B. If the maximal value is negative or 0, the process is stopped and the division from C to A and B is completed.

Some variants of the process of dividing one cluster into two have been discussed in (Kaufman and Rousseeuw, 1990, Chapter 6). For example, the test function defined in equation (7.25) can be switched to

$$D(A\backslash\{\mathbf{x}\}, B \cup \{\mathbf{x}\}).$$

When a data point maximizes the above function, then it may be moved from A to B. A possible stopping rule is that $D(A_{new}, B_{new})$ no longer increases.

Since the algorithm DIANA uses the largest dissimilarity between two objects in a cluster as the diameter of the cluster, it is sensitive to outliers. Other techniques for splitting a cluster and examples of the algorithm have been presented in (Kaufman and Rousseeuw, 1990).

7.4.7 DISMEA

DISMEA, presented by Späth (1980) based on a divisive method proposed in Macqueen (1967), is a divisive hierarchical clustering algorithm that uses the k-means algorithm to subdivide a cluster into two. The divisive method produces a hierarchy that consists of n levels, where n is the size of the given data set. Starting with the whole data set, at each successive step, the cluster with the largest sum of squared distances (SSD) is divided into two clusters. This process is continued until every cluster contains a single data point.

More specifically, for a given data set D with n objects, the first step of the divisive method is to find a bipartition C_1, C_2 of D (i.e., $C_1 \neq \Phi, C_2 \neq \Phi, C_1 \cap C_2 = \Phi$, and $C_1 \cup C_2 = D$) such that the objective function

$$F(C_1, C_2; D) = \sum_{i=1}^{2} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \mu(C_i)\|^2 \tag{7.26}$$

is minimized, where $\mu(C_i)$ is the centroid of cluster C_i, i.e.,

$$\mu(C_i) = \frac{1}{|C_i|} \sum_{\mathbf{x} \in C_i} \mathbf{x}.$$

At the succeeding steps, the cluster with the largest SSD is selected to be divided, where the SSD for a given cluster C is defined as

$$E(C) = \sum_{\mathbf{x} \in C} \|\mathbf{x} - \mu(C)\|^2,$$

where $\mu(C)$ is the centroid of cluster C.

For example, let $C_1, C_2, \ldots, C_j$ $(j < n)$ be the clusters at a step. Then the next step is to divide C_{j_0} if

$$E(C_{j_0}) = \max_{1 \leq s \leq j} E(C_j).$$

One possible way to find the optimal bipartition is to examine all the $2^{|C|-1} - 1$ possible bipartitions and find an optimal one. However, this is impractical when the size of the cluster to be subdivided is large. Another approach is necessary to find an optimal bipartition.

Instead of examining all possible divisions, the algorithm DISMEA uses the k- means algorithm to subdivide a cluster into two. In practice, the maximum number of clusters k_{max} $(\leq n)$ is specified in the algorithm DISMEA. The FORTRAN code for the algorithm DISMEA and some examples have been presented in (Späth, 1980).

7.4.8 Edwards and Cavalli-Sforza Method

Edwards and Cavalli-Sforza (1965) have suggested a divisive hierarchical algorithm by successfully splitting the objects into two groups to maximize the between-groups sum of squares. In this algorithm, the cluster density is measured by the variance, i.e., the within-cluster sum of squares divided by the number of points. At the beginning of the algorithm, the whole data set is divided into two groups according to the criterion mentioned above.

Then each of the two groups will be split into two groups according to the same criterion. One then continues splitting until each cluster contains only one point.

The technique adopted by Edwards and Cavalli-Sforza (1965) to split a group of data points into two subgroups is to enumerate all possible bipartitions and choose the one that minimizes the within-group sum of squares. Let D be a set of n data points. Then there are $2^{n-1} - 1$ different ways to divide D into two clusters. More specifically, let A_i, B_i be a partition of D for $i = 1, 2, \ldots, 2^{n-1} - 1$, i.e., $A_i \cup B_i = D$, $A_i \cap B_i = \Phi$, and $A_i \neq \Phi$, $B_i \neq \Phi$. Then the within-cluster sum of squares of the partition (A_i, B_i) is

$$WSS_i = \frac{1}{|A_i|} \sum_{\mathbf{x},\mathbf{y} \in A_i} \|\mathbf{x} - \mathbf{y}\|^2 + \frac{1}{|B_i|} \sum_{\mathbf{x},\mathbf{y} \in B_i} \|\mathbf{x} - \mathbf{y}\|^2 \tag{7.27}$$

for $i = 1, 2, \ldots, 2^{n-1} - 1$.

The best partition is (A_{i_0}, B_{i_0}) such that

$$WSS_{i_0} = \min_{1 \le i \le 2^{n-1}-1} WSS_i.$$

Edwards and Cavalli-Sforza (1965) also pointed out that the best partition may not be unique in some cases. For example, if the distance between any two data points is the same, say, d, then splitting n points into two clusters A, B such that $|A| = r$, $|B| = n - r$ will give a within-cluster sum of squares of

$$\frac{1}{r} \cdot \frac{1}{2} r(r-1)d^2 + \frac{1}{n-r} \cdot \frac{1}{2}(n-r)(n-r-1)d^2 = \frac{1}{2}(n-2)d^2,$$

which is independent of r.

For the Edwards and Cavalli-Sforza method, a major difficulty is that the initial division requires an examination of all $2^{n-1} - 1$ bipartitions, where n is the size of the original data set. This will take an enormous amount of computer time (Gower, 1967). Scott and Symons (1971a) suggested a refined algorithm that limits the consideration to $(2^d - 2)\binom{n}{d}$ partitions, where n is the size of the data set and d is the dimensionality of the data set. The improved algorithm is based on the results by Fisher (1958). Regarding the minimum variance partition for the univariate case, i.e., $d = 1$, Fisher (1958) defines a contiguous partition to be such that if x_i and x_j $(x_i \le x_j)$ belong to the same group, then every object x_k with $x_i \le x_k \le x_j$ also belongs to the group, and he proved that the optimal partition is contiguous.

For $d \ge 2$, Scott and Symons (1971a) generalized the definition of a contiguous partition into k groups to be such that if each member of a set of data points belongs to the same group, then every data point in the convex hull (Barber et al., 1996) of the set also belongs to the group. They also generalized Fisher's result as follows:

1. The minimum variance partition is contiguous.
2. For $d \ge 2$, the two groups of the minimum variance partition are separated by a $(d-1)$-dimensional hyperplane containing d of the data points.

Thus for each of the $\binom{n}{d}$ choices of d data points, there are $2^d - 2$ $(d \ge 2)$ possible assignments of the d points in the hyperplane into two groups. Since it is simple to decide

on which side of the hyperplane each of the remaining $n - d$ points will lie, there are a total of $(2^d - 2)\binom{n}{d}$ partitions that should be considered.

Note that, if $d \geq 2$, then the same partition will be considered many times. For example, when $d = 2$, the same partition will be considered twice. In fact, it has been shown that the number of distinct contiguous partitions is given by (Harding, 1967)

$$v_d(n) = \sum_{i=1}^{d} \left(\begin{array}{c} n \\ i \end{array} \right).$$

7.5 Summary

Comprehensive discussions of hierarchical clustering methods can be found in (Gordon, 1987) and (Gordon, 1996). Techniques for improving hierarchical methods are discussed in (Murtagh, 1983), and (Murtagh, 1984a) discussed the complexities of some major agglomerative hierarchical clustering algorithms. A review of applying hierarchical clustering methods to document clustering is given in (Willett, 1988). Other discussions are provided in (Hodson, 1970), (Lance and Williams, 1967a), and (Lance and Williams, 1967c).

Posse (2001) proposed a hierarchical clustering method for large datasets using the MST algorithm to initialize the partition instead of the usual set of singleton clusters. Other hierarchical clustering algorithms can be found in (Rohlf, 1970) and (Day and Edelsbrunner, 1984).