6

Network analysis

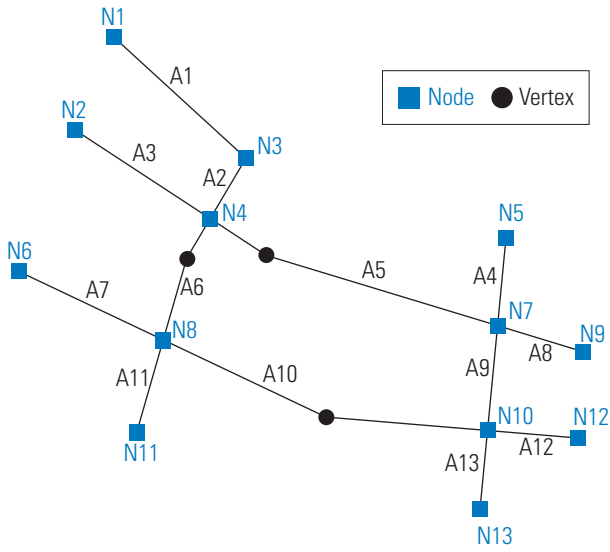
6.1 Introduction

The previous chapter dealt with combining information from different data layers, with one objective being to define connections between objects. This chapter is concerned with connections between places within networks. In particular, methods for the characterization of network complexity and for ascertaining shortest paths between start and end points are detailed. In Section 4.3, approaches for measurement of line lengths were discussed while information on connectivity of vector features was considered in Section 2.2.3. These can be put to use in the analysis of networks. For instance, finding the shortest route between one place and another through a road network is an example of a network analysis approach that necessitates measurement of distance and information on the connections between arcs (representing, for example, roads). Many real-world applications make use of such approaches—the determination of optimal routes for emergency vehicles is one such case. In this chapter some key ways to explore networks are detailed. The account is selective, but will provide a basis on which readers can build.

6.2 Networks

Figure 6.1 shows the constituent parts of a simple network. These include arcs, vertices (representing a change in direction of the arc), and nodes at the end of each arc and connecting different arcs.

Each segment of a network is associated with an impedance factor. In many cases this is simply the distance between nodes, but some alternative factor such as travel cost or time might be used instead. Before network analysis can proceed it is necessary to determine the impedance factors. For a road network, penalties on left turns or right turns (depending on the side of the road on which vehicles drive) might be imposed and u-turns, for example, may or may not be allowed.



Arc	From node	To node	Length
1	1	3	1.358
2	3	4	0.529
3	2	4	1.230
4	5	7	0.665
5	4	7	2.335
6	4	8	0.991
7	6	8	1.204
8	7	9	0.674
9	7	10	0.797
10	8	10	2.588
11	8	11	0.728
12	10	12	0.687
13	10	13	0.598

Figure 6.1 Synthetic road network. Arc numbers are prefixed by 'A' and node numbers by 'N'.

The initial focus of this section is on simple summaries of networks, in terms of how connected places are or how complex are the networks which connect these places.

6.3 Network connectivity

The connectivity of a network can be represented using the connectivity matrix (Taafe *et al.*, 1996). The connectivity matrix is a square matrix that contains the arc labels as its column and row headings. The matrix indicates those nodes that are connected by an arc (assigned a value of 1) and those that are not (given 0). The connectivity matrix for the network shown in Figure 6.1 is given in Table 6.1. In this case, nodes 1 and 3, for example, are connected by an arc—that is, the value for column 3 and row 1, or for column 1 and row 3, is 1, indicating a connection.

The matrix in Table 6.1 includes a final column that is the sum of all the row elements and it indicates the number of ways in which a node can be reached in one step from other nodes. The maximum number of possible ways (4) is for nodes 4, 7, 8, and 10. The connectivity matrix, indicated by $\mathbf{C}^1$, in Table 6.1 refers to nodes that are directly connected by arcs—it is termed 'first order'. This idea can be extended to the case of nodes that are connected by two arcs with a further node in between—this is termed 'second order' (and the matrix is indicated by $\mathbf{C}^2$). The matrix in the second-order case is obtained by multiplying $\mathbf{C}^1$ with itself (given by $\mathbf{C}^1\mathbf{C}^1$). Matrix multiplication is illustrated below, but further supporting material is given in Appendix A if clarification is required. To multiply a matrix by itself the matrix is multiplied by the transpose (as defined in Appendix A) of itself. Note that, in this case, the result of

Table 6.1 Connectivity matrix for the network shown in Figure 6.1 = matrix $\mathbf{C}^1$

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	Sum
1	0	0	1	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	1	0	0	0	0	0	0	0	0	0	1
3	1	0	0	1	0	0	0	0	0	0	0	0	0	2
4	0	1	1	0	0	0	1	1	0	0	0	0	0	4
5	0	0	0	0	0	0	1	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	1	0	0	0	0	0	1
7	0	0	0	1	1	0	0	0	1	1	0	0	0	4
8	0	0	0	1	0	1	0	0	0	1	1	0	0	4
9	0	0	0	0	0	0	1	0	0	0	0	0	0	1
10	0	0	0	0	0	0	1	1	0	0	0	1	1	4
11	0	0	0	0	0	0	0	1	0	0	0	0	0	1
12	0	0	0	0	0	0	0	0	0	1	0	0	0	1
13	0	0	0	0	0	0	0	0	0	1	0	0	0	1

matrix multiplication is the same if the matrix is multiplied by itself and not by its transpose, as the connectivity matrix is symmetric, for example the value for column 2, row 3 is the same as the value for column 3, row 2.

To generate the top left value in the new matrix $\mathbf{C}^2$ take each element in that row (from left to right) from the first matrix (i.e. $\mathbf{C}^1$) and multiply it by each element in that column (from top to bottom) of the second matrix (since we are multiplying one matrix by itself, the first and second matrix are the same). In words, with the column given first and then the row (where ID is the node label given in Table 6.1 and ID(1,1) is 0 while, for example, ID(1,3) is 1):

$$\mathrm{ID}(1,1)\times\mathrm{ID}(1,1)+$$

$$\mathrm{ID}(2,1)\times\mathrm{ID}(1,2)+$$

$$\mathrm{ID}(3,1)\times\mathrm{ID}(1,3)+\cdots$$

and so on until the end of the column and row. At that point add together all the multiplied values. This final summed value is written to cell 1,1 in the new matrix $\mathbf{C}^2$. Note that the row number steps up (increments) in the first matrix while the column number increments in the second.

Next move on to column 2:

$$\mathrm{ID}(1,1)\times\mathrm{ID}(2,1)+$$

$$\mathrm{ID}(2,1)\times\mathrm{ID}(2,2)+$$

$$\mathrm{ID}(3,1)\times\mathrm{ID}(2,3)+\cdots$$

and so on until the end of the column and row. At that point add together all the multiplied values. This final value is written to cell 2,1 in the new matrix $\mathbf{C}^2$.

When results for all columns have been processed, move onto column 1, row 2:

$$ID(1,2)\times ID(1,1)+$$
$$ID(2,2)\times ID(1,2)+$$
$$ID(3,2)\times ID(1,3)+\cdots$$

and so on until the end of the column and row. At that point add together all the multiplied values. This final value is written to cell 2,2 in the new matrix $\mathbf{C}^2$. See Appendix A if further examples of matrix multiplication are required.

This process is completed for all columns and rows. The end result is shown in Table 6.2.

Note that in the case of, for example, connectivity between node 4 and itself, the value of 4 in $\mathbf{C}^2$ indicates movement from node 4 to node 2, 3, 7, or 8 and back to node 4. The matrix for order 3, $\mathbf{C}^3$, is obtained by multiplying the matrices $\mathbf{C}^1$ and $\mathbf{C}^2$ together. The number of meaningful **C** matrices is determined by the diameter of a network. The diameter of a network is defined as the maximum number of steps needed to move from any to node to any other node in the network using the shortest possible route (Chou, 1997). In the case of the network in Figure 6.1, the network diameter is five—that is, five steps are necessary to move from node 1 to node 12 or node 13. Therefore, in this case a matrix of order 5 is meaningful, but this is not the case for any higher order. Adding the entries of each cell in each of the **C** matrices ($\mathbf{C}^1$ to $\mathbf{C}^5$ for the example) gives the total accessibility matrix, or *T* matrix, which indicates the number of ways to move between one node and another in a given number of steps (five in the example) or less (Taafe *et al.*, 1996). Taafe *et al.* (1996) demonstrate the application of the *T* matrix for identifying the new link that would most markedly increase the

Table 6.2 Connectivity matrix for order 2 = matrix $\mathbf{C}^2$

ID	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	0	0	1	0	0	0	0	0	0	0	0	0
2	0	1	1	0	0	0	1	1	0	0	0	0	0
3	0	1	2	0	0	0	1	1	0	0	0	0	0
4	1	0	0	4	1	1	0	0	1	2	1	0	0
5	0	0	0	1	1	0	0	0	1	1	0	0	0
6	0	0	0	1	0	1	0	0	0	1	1	0	0
7	0	1	1	0	0	0	4	2	0	0	0	1	1
8	0	1	1	0	0	0	2	4	0	0	0	1	1
9	0	0	0	1	1	0	0	0	1	1	0	0	0
10	0	0	0	2	1	1	0	0	1	4	1	0	0
11	0	0	0	1	0	1	0	0	0	1	1	0	0
12	0	0	0	0	0	0	1	1	0	0	0	1	1
13	0	0	0	0	0	0	1	1	0	0	0	1	1

connectivity of a road network. Tools like the *T* matrix are potentially very useful aids to transport network development. The following section details some single-value summaries of network characteristics.

6.4 Summaries of network characteristics

Several simple summaries of network characteristics exist. These include the γ (gamma) index and the α (alpha) index (Taafe *et al.*, 1996), both of which are introduced below. A better connected network has larger values of γ and α (Chou, 1997) and together they provide a useful summary of network complexity and connectivity.

For a given number of nodes, more arcs indicate greater connectedness. The minimum number of arcs (or links), *l*, needed to connect *n* nodes is given by:

$$l_{\min} = n - 1 \tag{6.1}$$

With a minimally connected network, removal of any one arc will result in two unconnected networks—that is, there are no loops or circuits in the network. For the network in Figure 6.1, the number of nodes is the same as the number of arcs—there is a circuit comprising arcs 5, 6, 10, and 9. It is therefore not minimally connected, but it would be if arc 5 or arc 10 was removed from the circuit.

There is a range of measures of network characteristics, such as complexity of a network or the degree to which it is connected. The γ index is one summary of network complexity. It is the ratio of the number of links (arcs) in a network to the maximum number of links possible. For a planar graph with *n* nodes, the maximum number of links is given by $3(n-2)$. A graph is a set of nodes connected by arcs; a planar graph has no intersecting arcs—the intersections in Figure 6.1 are represented by nodes which connect separate arcs and so it is a planar graph. In non-planar graphs, such as three-dimensional air transport networks, the maximum number of links is $^{n(n-1)}/_{2}$ (Chou, 1997). The γ index for a planar graph is given by (Chou, 1997):

$$\gamma = \frac{l}{l_{\max}} = \frac{l}{3(n-2)} \tag{6.2}$$

where *l* is the number of links in the network and $l_{\max}$ is the maximum number of possible links (i.e. $3(n-2)$). The γ index can take values from 0 to 1 where small values indicate simpler networks with few links and larger values indicate more links, and therefore a better connected network (Chou, 1997).

The network illustrated in Figure 6.1 has 13 nodes and 13 lines (arcs), so $n=13$ and $l=13$. In this case, γ is calculated by:

$$\gamma = \frac{13}{3(13-2)} = \frac{13}{33} = 0.394$$

If five links were added to the existing set of links and nodes (the number of nodes remaining the same, e.g. adding a link between the existing nodes 3 and 7) the updated calculations would be:

$$\gamma = \frac{18}{3(13-2)} = \frac{18}{33} = 0.545$$

The larger value of γ than in the previous case reflects the larger number of connections.

An additional measure of network connectivity measures the number of circuits, c, that exist within a network. A circuit has a start node that is the same as the end node and it comprises a closed loop (Lee and Wong, 2000). In a minimally connected network (defined previously in this section) there are no circuits and the number of circuits can be calculated by subtracting the number of arcs required to form a minimally connected network from the observed number of arcs in the network. This is given by $l-n+1$ (recall that l is the number of arcs and n is the number of nodes). In the case of the network shown in Figure 6.1, $l=13$, which gives $13-13+1=1$, i.e. there is one circuit. The α index is the ratio of the number of circuits (c) to the maximum possible number of circuits in a network (c_{max}) (Chou, 1997). It is given by:

$$\alpha = \frac{c}{c_{max}} = \frac{l-n+1}{2n-5} \tag{6.3}$$

For the network in Figure 6.1, this gives $^1/_{21} = 0.048$.

Scott *et al.* (2006) review the γ index, amongst other measures, and present an additional measure for assessing the importance of particular links and for evaluating the connectivity of networks. Such measures may be useful, for example, in informing the design of new road networks or in altering those already in existence.

6.5 Identifying shortest paths

Identification of the shortest path between two places is a common aim in the analysis of networks. A widely used approach to identifying the shortest path between two places is the algorithm presented by Dijkstra (1959). The basis of this approach is that links, connected to a starting node, are selected that have the shortest path back to that starting node, and the algorithm is outlined below. Such an approach is much more efficient than an approach that simply calculates all possible routes and selects the shortest one. A worked example is given based on the network shown in Figure 6.2.

In the example, the objective is to identify the shortest path from node 1 to node 6. This is a straightforward task in this case, but in most real-world cases some automated procedure is necessary. With the initial step, represented using the 'Starting

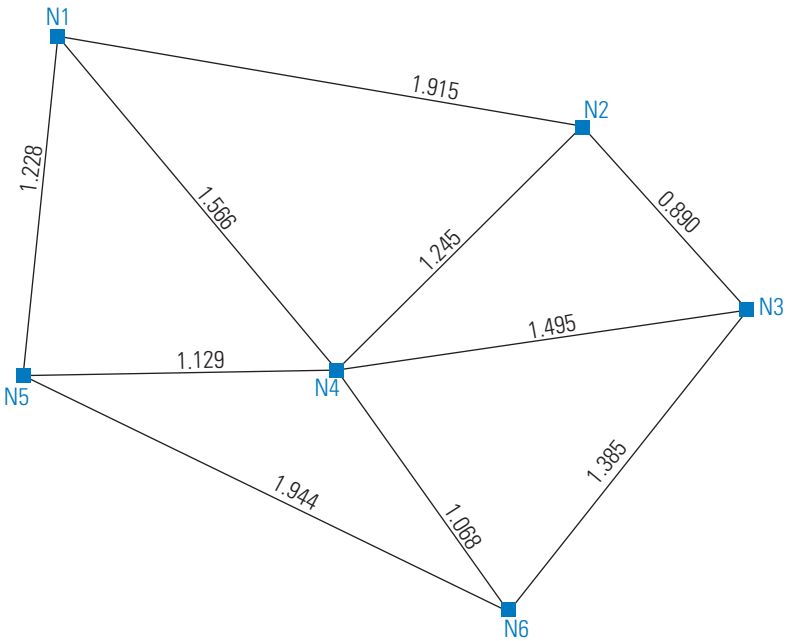


Figure 6.2 Network used for illustrating the shortest path algorithm, with the length of each arc indicated. 'N' indicates node.

table' in Table 6.3, each node has a distance from the source node that is infinity (∞) and the next node back from each node (parent) to the origin is blank. 'Included' indicates nodes which have comprised part of the shortest route to a node at any stage. Note that, using this approach, costs (e.g. distances) must be positive. The example is presented numerically, rather than graphically, as a key concern is to show exactly how the algorithm operates.

The description of the shortest path algorithm that follows is based on that given by Wise (2002) and makes reference to the format shown in Table 6.3:

- Select the non-included node with the shortest distance to the source.
- Include this node (i.e. replace 'N' in the final column with 'Y'). The distance from this node back to the source node is indicated by dist(n).
- Identify the nodes connected to node n that are still not included (they have an entry 'N' in their final column). The distance between each of these nodes (m) and the node n is given by d(nm).
- For each of the m nodes:
  if dist(n) + d(nm) < dist(m) then dist(m) = dist(n) + d(nm) and parent(m) = n
  that is, if there is a shorter route from the source node to node m via node n then the entry for node m is updated with the new total distance figure and the parent value for node m is set to n (the previous node along the path from the origin node). The algorithm is illustrated below with reference to the sub-tables that comprise Table 6.3.

Table 6.3 Starting table and table after each iteration of Dijkstra's shortest path algorithm

Starting table

N	Distance	Parent	Included
1	∞		N
2	∞		N
3	∞		N
4	∞		N
5	∞		N
6	∞		N

After iteration 1

N	Distance	Parent	Included
1	0	–	Y
2	1.915	1	N
3	∞		N
4	1.566	1	N
5	1.228	1	N
6	∞		N

After iteration 2

N	Distance	Parent	Included
1	0	–	Y
2	1.915	1	N
3	∞		N
4	1.566	1	N
5	1.228	1	Y
6	3.172	5	N

After iteration 3

N	Distance	Parent	Included
1	0	–	Y
2	1.915	1	N
3	3.061	4	N
4	1.566	1	Y
5	1.228	1	Y
6	2.634	4	N

After iteration 4

N	Distance	Parent	Included
1	0	–	Y
2	1.915	1	Y
3	2.805	2	N
4	1.566	1	Y
5	1.228	1	Y
6	2.634	4	N

After iteration 5

N	Distance	Parent	Included
1	0	–	Y
2	1.915	1	Y
3	2.805	2	N
4	1.566	1	Y
5	1.228	1	Y
6	2.634	4	Y

Firstly, the distance from node 1 is set to 0, as it is the source node. The 'Included' entry for node 1 is set to Y, to indicate it is part of the shortest path (obviously it must be as it is the source node!). Node 1 is connected to nodes 2, 4, and 5, and the distances from node 1 to these nodes are smaller than infinity and so replace the existing entries in the distance column. The parent nodes for nodes 2, 4, and 5 are set to 1. The result is the table 'After iteration 1' (where each iteration represents a single cycle in the process).

The shortest distance for any node that has not yet been included is that for node 5 (a distance of 1.228) and this node is now included. Node 5 is connected to two nodes down the path from node 1: nodes 4 and 6. The distance to node 4 via node 5 is larger than the distance already recorded for node 4, which is direct from node 1, so the entry for node 4 remains the same. The distance from node 5 to node 6 is 1.944 units. This is added to the distance from node 1 to node 5 (1.228), giving a total path length from node 1 via node 5 to node 6 of 3.172 and the parent node for node 6 is set to 5 (i.e. the connection back to node 1 is via node 5). This distance replaces the infinity value for node 6. This gives the table 'After iteration 2'.

The shortest distance for any non-included node is 1.566, for node 4, so this node is now included. Node 4 is connected to nodes 2, 3, and 6. The distance from node 1 to node 2 via node 4 is greater than the direct link between node 1 and node 2, so the entry for node 2 remains the same. The distance from node 1 to node 3 via node 4 is 1.566 + 1.495 = 3.061 and this value replaces the infinity value for node 3, for which the parent node is set to 4. The distance from node 1 to node 6 via node 4 is 1.566 + 1.068 = 2.634. This is smaller than the existing distance value for node 6, so the distance value is replaced and the parent node is changed to 4. The result is the table 'After iteration 3'.

The shortest distance for any node that has not yet been included as part of the shortest route is 1.915. This is the distance value for node 2 and node 2 is included at this stage. Node 2 is connected to nodes 3 and 4. The distance from node 1 to node 3 via node 2 is 1.915+0.890=2.805 and is smaller than the existing distance value for node 3, so the distance value for node 3 is replaced and the parent node changed to 2. Node 4 is already included so it is not considered. The end result is the table 'After iteration 4'.

The distance of 2.634, for node 6, is the smallest distance for a non-included node. Node 6 is included at this stage. Node 6 is connected to the remaining non-included node, node 3. The distance from node 1 to node 3 via nodes 4 and 6, however, is greater than the existing distance value so it remains the same. This is the final stage and results in the table 'After iteration 5'.

The distance values for each node are the shortest path distances to that node from node 1. For example, the shortest path from node 1 to node 4 is 1.566 units while the shortest path from node 1 to node 6 is via node 4 and is 2.634 units (1.566+1.068). Wise (2002) and Chang (2008) provide further worked examples of the use of the shortest path algorithm.

6.6 The travelling salesperson problem

There are many other methods for the analysis of networks that could be outlined. For instance, algorithms for solving the so-called 'travelling salesperson problem' (TSP) are often encountered in GIS textbooks. The TSP entails finding the shortest possible route that visits each of a set of points once only and returns to the starting point. The number of possible routes increases markedly with the number of locations that must be visited. Where there are n locations that must be visited, including the origin (e.g. the depot), the number of possible routes is given by $(n-1)!$, where ! indicates the factorial of a number. As an example, 3! is $3\times2\times1=6$. In practice, most trips can be taken in two directions and then the possible number of routes is given by $^{(n-1)!}/_{2}$ (Longley *et al.*, 2005a), so if there are eight locations to be visited, $n-1=7$ and $7!=7\times6\times5\times4\times3\times2\times1=5040$ and the number of possible routes is $^{5040}/_{2}=2520$. In practice, for a large number of places to visit, it is not feasible to identify the optimal route, but approaches exist to rapidly identify routes that may not be the best, but which approach the optimal route according to some criterion. Wise (2002) provides a summary of the TSP and possible approaches to it.

6.7 Location–allocation problems

A major class of problems for which network analysis tools provide a solution is the matching of resources to particular people or groups. The term 'location–allocation' refers to the location of facilities and the allocation of resources from a given facility

to particular locations. For example, we may wish to determine which depot may make deliveries of goods to which particular areas within a maximum travel time limit (using information on road distances and perhaps other impedance factors such as average travel time). Information on supply and demand levels may be used to determine the maximum catchment area of a facility (see Birkin *et al.* (1996) for a summary). Approaches to solving the location–allocation problem may be used to help ascertain if a service provider can meet the needs of a given area or if a new facility (e.g. a hospital or a retail outlet) might be needed or existing facilities expanded. Chang (2008) provides an introduction to location–allocation modelling with several examples.

6.8 Case study

As in most other chapters, this chapter concludes with a case study that can be conducted using standard GIS software. The data are provided on the book website so that

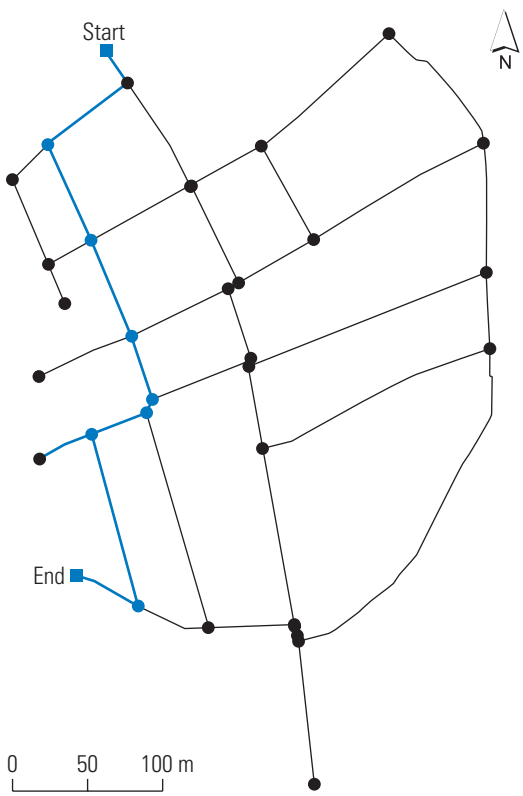


Figure 6.3 Shortest path between start and end point. The nodes of the network are given by circles and the start and end points by squares.

readers can explore the application of the selected methods. Guidance notes are also provided on the website to outline how the methods are applied in these particular cases.

The data used in this case study represent the street layout of the town of Aberystwyth, captured from historic maps as part of a project exploring the morphology (i.e. the shape) of various medieval town plans in England and Wales[1] (Lilley *et al.*, 2007). In this case study, the objective was to identify the shortest path between the start and end points indicated in Figure 6.3. Such problems are common, for example any road user may wish to identify the shortest route between their starting point and their destination. Information on toll costs, traffic density, and other issues that may affect the choice of route can easily be taken into account. ArcGIS™ Utility Network Analyst offers a routine to compute shortest paths along networks and this software was used to identify the shortest path, which is indicated by a heavy line in Figure 6.3.

With an efficient approach like Dijkstra's algorithm, it is possible to rapidly compute shortest paths while taking into account factors such as traffic jams where such information is available.

Summary

This chapter introduced a variety of approaches to characterizing networks which may have application in, for example, transportation planning contexts. An algorithm for identifying shortest paths was also outlined. Brief summaries of some important network analysis problems were also provided. This chapter provides a foundation for understanding a major class of approaches for the analysis of vector data, and it presents the basic principles that are then foundations of a range of methods frequently applied to solve real-world problems. The suggested further reading provides a way forward if these kinds of approaches are of interest.

Further reading

Wise (2002) details some algorithms for network analysis. That book provides a good starting point for readers wanting to know more about GIS algorithms as a whole. Further accounts of network analysis are provided by **Chou (1997)**, **Lee and Wong (2000)**, and **Chang (2008)**. **Taaffe *et al.* (1996)** provide an in-depth account of transportation geography that provides the context for the material covered in this chapter.

➔ This chapter was concerned with line features and their analysis; the analysis of point patterns (i.e. sets of point 'events') is the subject of the following chapter.

1 http://www.qub.ac.uk/urban_mapping/