Doing Social Network Research

FOUR

Social systems and data structures: Relational ties and actor attributes

In Chapter 3, I wrote that there was no unique network abstraction to be universally applied. We need to think carefully about how best to represent the features of the particular social system or social context that we are studying, and adapt our network conceptualization accordingly.

In this chapter, I present different types of data structures applicable to the types of research designs and questions in Chapter 3. The research question obviously has implications for the appropriate data to be collected. A given data structure is integral to the design of the study and to the method of data collection.

Following this chapter, I will take up general issues for network data collection in Chapter 5, and different contexts in which network data may be collected in Chapter 6. The data structures presented in this chapter can be used for network visualization (Chapter 8) and for subsequent analysis (Chapter 9).

Qualitative and quantitative data

Much network research has a strong quantitative aspect, but that does not mean that qualitative methods are not applicable. I will discuss qualitative network studies in more detail in Chapter 6. The current chapter concentrates on the structure of numerical network data, but that is not to privilege quantitative methods. Indeed, qualitative research based on careful interviews, if it is network-based, will need to extract relational information that can then be translated into the data structures of

this chapter. This will be necessary, for instance, if the qualitative researcher wishes to visualize the networks using standard software. So, if you are a qualitative researcher and prefer to exclude quantitative inference, you may still want to produce the binary edge lists or matrices described below. Because I talk of data, matrices and vectors, that does not mean that this chapter is irrelevant for qualitative research.

Qualitative network research often uses an egonet design, where ego is the interviewee from which qualitative data is obtained. The structure of egonet data is described at the end of this chapter. It is important for the qualitative researcher to understand this data structure if an explicit network framework is to be used, even when the original data is not numeric. For network mixed methods approaches, where qualitative and quantitative research are integrated, all the data structures of this chapter are potentially relevant.

Some network notation

Not everyone is keen on mathematical notation, so I will keep it to a minimum. Nevertheless, it is helpful for this and following chapters to have some simple mathematical descriptors of the network structures we examine. My notation is presented in Box 4.1. (Note that there is no one standard notation, and different authors have different practices.)

From this point, I will start to use terminology such as tie or attribute *variables*. The use of the term *variable* is again not intended to privilege a quantitative, or more particularly, a statistical point of view, although some but certainly not all of the methods described in Chapter 9 are statistical. A tie is possible between pairs of actors, but need not be present. Hence the presence of a tie can be said to vary across pairs of actors. Similarly, an actor may or may not have a particular attribute (e.g., be female), so that attribute can be said to vary across actors. At this point, I intend no more than this in the use of the term *variable*.

Often, but not always, tie variables are binary, so that they take the value of 1 or 0 depending on whether the tie is observed or not. As usual, a variable needs to be distinguished from the value it may actually take. For instance, *age* may be a variable in a study, and an actor, John, may have an age of 37. The variable here is *age* and its value for John happens to be 37.

BOX 4.1

Some network notation

- X_{ij} denotes a network tie variable between actor i and actor j. If the network is directed, X_{ij} denotes an arc variable from i as sender to j as receiver. For binary networks, $X_{ij} = 1$ if the edge or arc is observed, and $X_{ij} = 0$ if not. If the tie is *valued*, then

X_{ij} may take a value in whatever range is permissible. To differentiate the variable from the value it may take, sometimes I write $X_{ij} = x_{ij}$ to signify that, in the observed data, the tie variable X_{ij} takes the value x_{ij} for the pair of actors (i, j). In a unipartite network, it is usual (though not universal) for self-ties from a node to itself to be excluded, so that x_{ii} is forced to be zero.

- Y_i denotes an actor attribute variable for the actor i. To differentiate the variable and its observed value, sometimes I write $Y_i = y_i$.
- The set of all the tie variables (i.e., all of the X_{ij}) may be written as X, which then represents a variable that describes the whole network (not just for an individual (i, j) pair). Sometimes I write x to represent actual network data, so that x is the collection of all observed values $x_{ij.}$
- Suppose there are k types of relational ties in a multiple or multiplex network study. Then I write X_{ijm} to denote a tie variable between actor i and actor j on the mth type of relationship.
- In a longitudinal network study, I write X_{ijt} to denote a tie variable between actor i and actor j at time t.

Relational data structures

Now let me describe various network data structures. An understanding of the data structures will help you appreciate the possibilities of the designs in Chapter 3. I then introduce actor attribute variables. I postpone egonet data structures until late in the chapter, because they typically include attribute variables.

Many social science researchers will be used to entering the data into a spreadsheet or perhaps a favorite statistical package. While this is possible for network data, it is quite common to enter the data into simple text files. Most network analytic software can read in text files in a straightforward way, although the precise format depends on the program.

Whole network data

I begin with a whole network research design because this is so often the standard way in which network data is described. Recall that in a whole network we have a subset of actors defined within the boundary of the study. The data then comprises relational ties among the actors. In short, we represent data as observations x_{ij} among pairs of actors.

To begin, we give each of the n actors an identification number (ID), usually sequentially from 1 to n. Then there are two standard and equivalent ways to enter the network data.

- An *adjacency matrix* (sometimes called a *sociomatrix* for social network data): Here each of the rows and columns represent the actors, and the cell in row i and column

j takes the value x_{ij}. For a binary network, then, the adjacency matrix is a matrix of 0s and 1s representing the absence or presence of a tie.

- An *edge list*: This is simply a list of all the edges present in the network (i.e. where $x_{ij} \neq 0$) in three columns, describing the sender i, the receiver j and the value of the tie x_{ij}. If the network is binary, the third column x_{ij} is often dropped because it is 1 for all rows in the list.

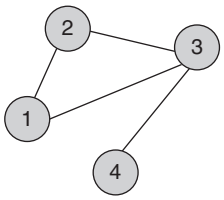
Figure 4.1 depicts a small example of an undirected four-node network, with both its adjacency matrix and edge list. Figure 4.1(a) visualizes the network with the nodes labelled by numbers. Figure 4.1(b) shows the adjacency matrix where the first row and column in the table are headings for the node numbers. So, for instance, cell (1,1) is 0, because self-ties are not permitted; whereas cell (1,2) is 1 because there is a tie between nodes 1 and 2. Because $x_{ii} = 0$ (no self-ties), the diagonal of the matrix is forced to be 0. Notice that because the network is undirected the matrix is symmetric (the top right of the matrix above the diagonal is the same as the bottom left below the diagonal).

Figure 4.1(c) presents the same matrix without the row and column headings of node numbers. This is how it is usually presented in network data, where the first column is assumed to apply to node 1 and so on (it is for this reason that it is often convenient to label the node IDs sequentially from 1 to n). Figure 4.1(d) provides the edge list. Notice that because the network is undirected, the order of i and j does not matter. It is common for the full adjacency matrix to be included as in (b) and (c) even though it is symmetric (so some of the information is redundant), but it is not usual in the edge list to repeat the same undirected tie for i and j and for j and i. In other words, for an edge list, if there is a tie between nodes 1 and 2, it is entered once, and not repeated for nodes 2 and 1. Notice that in the edge list in Figure 4.1(d), the last column x_{ij} has been dropped because this is a binary network.

Figure 4.2 now presents an example where there are values on the edges. In this case, the ties may be weighted 1, 2, or 3. Now the adjacency matrix is no longer binary and the edge list includes the third column x_{ij} to indicate the values on the edges. The thickness of the lines in the visualization indicates the weights.

Figure 4.3 depicts an example of a binary directed network. Note now that the adjacency matrix is not symmetric and the order of actors in the edge list is important. (Strictly speaking, it is now an *arc list*, although the term *edge list* is still often used for directed networks.) We can see from Figure 4.3 (a) that there is an arc from node 2 to 1 but not from 1 to 2. This is represented in the matrix as a 1 in cell (2,1) but a 0 in cell (1,2). In the edge list, there is an entry for '2 1' but not for '1 2'. The presence of an arc from i to j as well as from j to i means that there are *reciprocated* or *mutual* ties between those two actors. We can see this with ties from actors 2 to 3 and from 3 to 2, with both cells in the matrix entered as 1 and both arcs entered in the edge list.

Commonly, network analysis software will require an adjacency matrix or an edge list to enter network data. The edge list seems a more efficient way to enter the data



(a)

	1	2	3	4
1	0	1	1	0
2	1	0	1	0
3	1	1	0	1
4	0	0	1	0

(b)

0 1 1 0
1 0 1 0
1 1 0 1
0 0 1 0

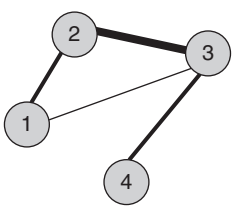
(c)

1 2
1 3
2 3
3 4

(d)

Figure 4.1 Undirected binary network data structure

(a) visualization; (b) and (c) adjacency matrices; (d) edge list.



(a)

0 2 1 0
2 0 3 0
1 3 0 2
0 0 2 0

(b)

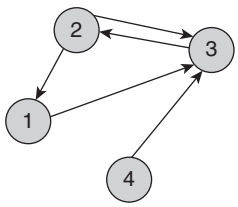
1 2 2
1 3 1
2 3 3
3 4 2

(c)

Figure 4.2 Undirected valued network data structure

(a) visualization; (b) adjacency matrix; (c) edge list.

because, of course, by definition it does not require any 0s to be entered, whereas in the matrix both 0s and 1s are included. However, it is easy to see that certain features of networks can be extracted from the adjacency matrix. In a binary undirected network, the degree of each actor is the sum across the rows (or the columns). For those who like mathematical formulae, the degree for actor i then is simply $\sum_{j=1}^{n} x_{ij}$.



(a)

	1	2	3	4
1	0	0	1	0
2	1	0	1	0
3	0	1	0	0
4	0	0	1	0

(b)

0 0 1 0

1 0 1 0

0 1 0 0

0 0 1 0

(c)

1 3

2 1

2 3

3 2

4 3

(d)

Figure 4.3 Directed network data structure

(a) visualization; (b) and (c) adjacency matrices; (d) edge list.

Again for a binary undirected network, the total number of edges L is the sum of all the cells in the matrix, divided by 2 for an undirected network because the top right of the matrix is the same as the bottom left – in short, just count all the 1s in the matrix and divide by 2 (i.e., $L = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}$).

The number of cells in the matrix is obviously n^2, but n of these are along the diagonal and forced to be 0, so cannot be edges. Because the cells below and above the diagonal are symmetric, only half of the non-diagonal cells represent distinct data, so the largest possible number of edges is $(n^2 - n)/2 = n(n - 1)/2$. Hence, the density of the observed network – the proportion of observed ties to possible ties – is $2L/n(n–1)$, as noted in Chapter 2.

In Figure 4.1(b), as n =4, we have $n(n - 1)/2 = 6$, which means there are six cells above (or below) the diagonal, as can readily be seen from the figure. As four of these contain 1s, the density is $4/6 = 0.67$.

For directed networks, on the other hand, row i in the adjacency matrix represents the choices of network partners made by i as sender, so that the sum of the row is the out-degree of $i\,(\sum_{j=1}^{n} x_{ij})$. The sum of the i-th column is the in-degree of i ($\sum_{j=1}^{n} x_{ji}$). Now there is no symmetry, so the total number of arcs is $L = \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}$ and the density is $L/n(n–1)$.

So just by looking at Figure 4.3(c), we can see that the out-degree of node 1 is 1 (i.e. the number of 1s in the first row), and the in-degree is also 1 (the number of 1s in the first column). Node 3 has an in-degree of 3 (the third column). Here the number of possible ties (non-diagonal cells in the matrix) is $n(n–1) = 12$. There are five arcs in this network, so the density is 5/12.

If you have some basic programming skills, it is not difficult to convert a binary edge list into an adjacency matrix. Box 4.2 describes a simple algorithm to do this. If you do not have programming skills, do not worry: if ever you need to do this, there is software that will do it for you (Chapter 9).

BOX 4.2

Creating an adjacency matrix from an edge list

1. Determine the number of nodes n and check that the nodes are numbered from 1 to n in the edge list (change the coding of the node IDs if not.)
2. Determine the number of edges L in the edge list.
3. Create an $n \times n$ matrix G with cells $G(i, j) = 0$.
4. Read in the edge list E as a $2 \times L$ matrix.
5. Loop though the L rows of E.
   - For each row r in E, set $i = E(r, 1)$ and set $j = E(r, 2)$.
   - Set $G(i, j) = 1$.
   - (If the network is undirected) Set $G(j, i) = 1$.
6. Finish when the loop in step 5 is complete.

There is an additional method of recording binary network ties – that of a *node list* or an *adjacency list*. In this method, the 'focal' node is placed in a first column and then every node connected to it is listed in subsequent columns. A new row presents a new focal node. The node list method is convenient for data entry, but usually needs to be converted into an edge list or adjacency matrix for entering into network software.

Multiplex networks

In a multiplex network study, there are several different types of relational ties on the same set of nodes. For instance, an organizational network study might examine collaboration, trust and friendship among a set of managers (e.g., Rank et al., 2010).

This case is a simple extension of the whole network data structure, except that there are now multiple adjacency matrices or edge lists, one for each type of relational tie (X_{ijm} where m goes from 1 to k for k different types of relational ties). These may be entered into k different files, or in some cases, if required, the matrix can be 'stacked' into one $kn \times n$ matrix in the one file as described below for longitudinal networks. (The choice here often depends on how the data will be processed after entry, in particular the form in which different software tools require the data.)

Cognitive social structures

Cognitive social structure data is an extension of a whole network study, but actors within the network boundary are asked about not only their own ties but also their perception of the ties among all other actors. In this case there is one adjacency matrix or edge list for actors' reports of their own ties, but each actor also has his or her own adjacency matrix reporting knowledge of ties among other actors in the network.

If there are many actors in the network, this may be a demanding task for the participant. The number of possible ties for an undirected network is $n(n - 1)/2$, which increases with the square of n, so for large n there may be very many possible ties for participants to report on. It is perhaps for this reason that this design is not very commonly used. The data structure for this design is similar to a multiplex design, except that now X_{ijm} indicates actor m perceiving a tie or not between actors i and j.

Snowball samples

Figure 4.4 depicts a two-wave snowball sample design. The first wave of data collection involves starting from a seed set of actors (perhaps from a random sample of nodes) and collecting data on their network ties. Pattison et al. (2013) referred to the seed set as *zone 0* nodes: in Figure 4.4 they are represented by the dark grey nodes.

The ties from zone 0 nodes include ties among the seed set, but also to another set of nodes not in zone 0. In the figure, these are represented by the grey nodes (zone 1). The second wave of data collection involves ties from the zone 1 nodes. These identify additional ties among zone 1 nodes but also ties to new nodes not in zones 0 and 1: zone 2 nodes (light grey in Figure 4.4). Because this is a two-wave design, network data is not collect from the zone 2 nodes, so we do not have data among those nodes, or among other nodes beyond zone 2. If there were more waves, we would continue to collect network data from each zone of actors until we reach the requisite number of waves. The dotted lines in the figure represent the boundaries of the different zones. Notice that a full two-wave snowball sample necessarily reaches all nodes within geodesic distance 2 of a seed set node. More generally, if there are k waves, the snowball sample comprises all nodes within geodesic distance k of a seed set node.

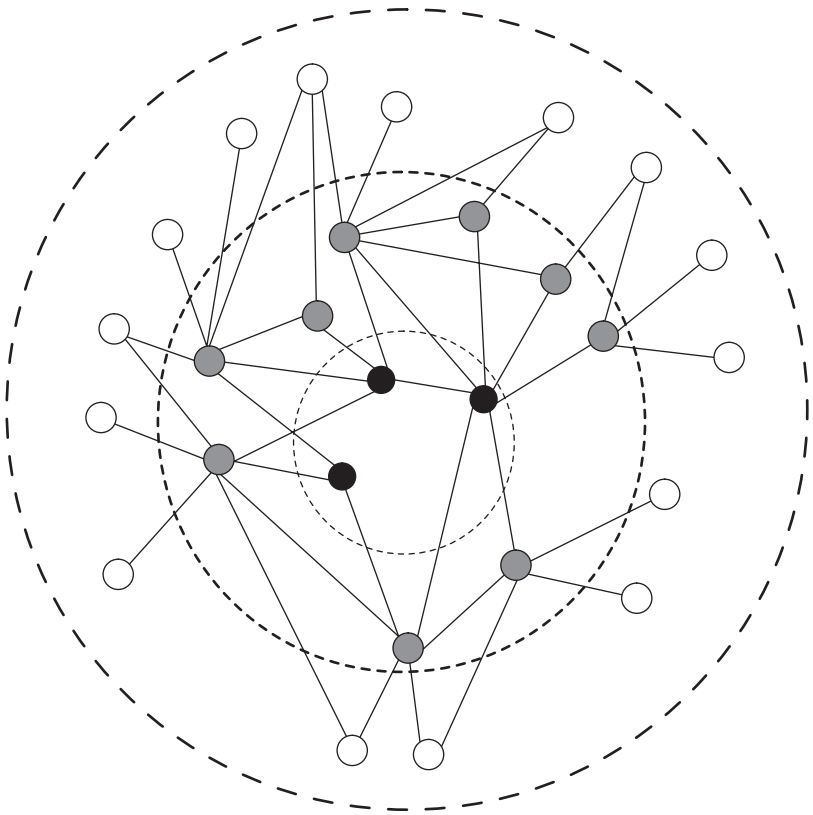


Figure 4.4 Snowball sample design

For a snowball sample, the data can still be presented as an edge list or an adjacency matrix, but an additional variable should be included for each node: namely, its zone. In data entry terms this may be included as an attribute variable, as discussed below when we come to attributes.

It is particularly important to note that in a snowball sample there are additional forced zeros and unobserved ties. For instance, by design in the data collection, it is impossible to have ties between zone 0 and zone 2, so they must be zeros in the adjacency matrix. None of the ties among zone 2 nodes are observed. Just as with the diagonal in a whole network adjacency matrix, they should not be included in any relevant calculations (such as the calculation of density). This shows that simply collecting snowball samples, entering data into an adjacency matrix and performing standard calculations (e.g., for density) without taking into account the snowball sample design is a serious error.

Figure 4.5 represents the adjacency matrix data for an undirected network (hence only the matrix above the diagonal is presented) showing what is known when a two-wave snowball is used. The nodes in the network are blocked and ordered by zones. Ties among zone 0 nodes and between zone 0 and 1 are observed, as are ties

between zone 1 and 2 nodes. By design there are no ties between zone 0 and zone 2 and remaining nodes (i.e. beyond zone 2), and between zone 1 and remaining nodes. All other ties are not observed.

	Zone 0 nodes	Zone 1 nodes	Zone 2 nodes	Remaining nodes
Zone 0 nodes	Observed	Observed	0	0
Zone 1 nodes		Observed	Observed	0
Zone 2 nodes			Not known	Not known
Remaining nodes				Not known

Figure 4.5 Regions of an adjacency matrix for a snowball sample design

The point of collecting a snowball sample is to make inferences applicable to the whole network using just the snowball data. Figure 4.5 shows that the zone structure of the nodes is a crucial part of the data, and if it is ignored incorrect results will follow. In Chapter 9, I will discuss some methods for the analysis of snowball samples.

The data structure for respondent-driven sampling has some similarities to snowball sampling but can differ in that some chains of referrals can be much longer than others. In this case, the data is best not conceptualized as a neat square matrix (albeit with gaps) as in Figure 4.5. You are more likely to have a tree-like structure, possibly with some intersecting branches (Heckathorn, 1997). Some branches in the tree will be short (or non-existent) when respondents do not recruit or recruit few partners. It is very important, however, to know who has recruited whom and for this to be recorded in your datafiles. Because RDS is typically applied to actor attributes rather than to network structure (the network structure is used as a sampling device to get at the attributes), this recruiting information will usually suffice in drawing sensible conclusions about the attribute distribution (Chapter 9).

Bipartite network data

Recall that a bipartite network has two types of nodes with ties between nodes of different types but not between nodes of the same type. Bipartite data often represents memberships, participation or attendance. For instance, bipartite data on company directors has ties from directors to their various company boards (reflecting membership), but no ties between companies or between directors.

Suppose that there are m and n nodes of the two different types. It may be more convenient to code IDs for the nodes from 1 to $m+n$, rather than have two separate codings for different node types. Bipartite data can then be represented in edge list or adjacency matrix form. In edge list form, it is probably best that the two columns (assuming binary data, i.e. no third column of tie strength x_{ij}) consistently represent the two different types of nodes (i.e. nodes of one type always in the first, and of the second type always in the second). The adjacency matrix is an $m \times n$ matrix and no longer square (unless it so happens that $m = n$), with no forced zeros on the diagonal.

Figure 4.6 presents an example of a bipartite network, together with its associated edge list and adjacency matrix. Suppose this is a network of two people (nodes 1 and 2) attending three events (nodes 3, 4, 5). The first row of the 2 × 3 matrix in Figure 4.6(c) represents the attendances by person 1 (i.e., at all three events), while the second row shows that person 2 did not attend event 4.

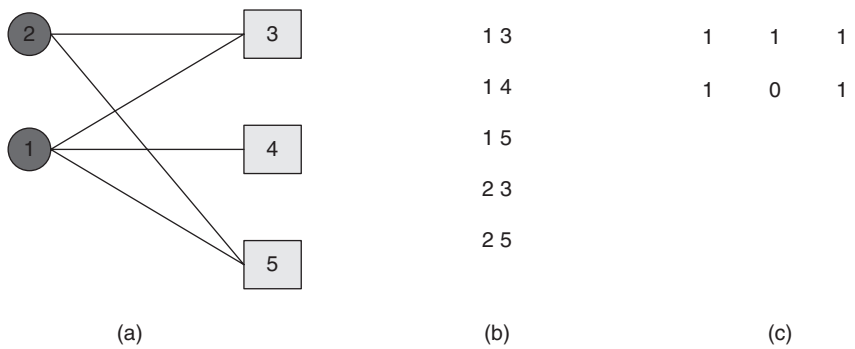


Figure 4.6 Bipartite network data structure

(a) visualization; (b) edge list; (c) adjacency matrix

Again, many simple network properties can be calculated from the matrix. The sums of the rows still constitutes a degree distribution as for unipartite networks, but here each row sum indicates how many events were attended be each actor (3 for person 1, and 2 for person 2). The sum of each columns shows how many people attended each event (2 for events 3 and 5, and 1 for event 4). There are no forced zeros in this matrix, so the total possible number of ties is the number of cells ($2 \times 3 = 6$). As there are five 1s in the matrix, the density of the bipartite network is 5/6.

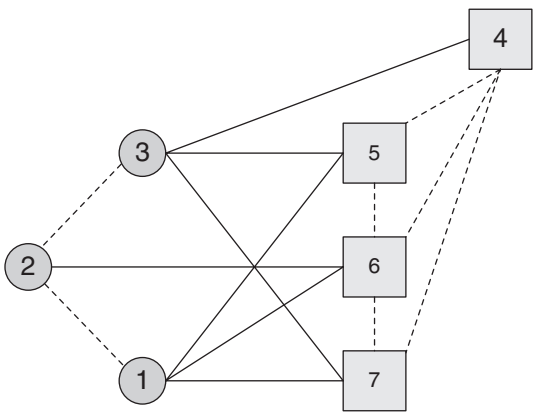
It is relatively rare for a study to collect k-partite data with $k > 2$ – that is, with k types of nodes. Mische (2008) provided an example of tripartite data, when she studied a Brazilian political movement with three types of nodes: political activists, organizations and events. Activists could be members of (multiple) organizations and attend events; organizations could be officially represented at events. Formally, in this case there is a three-way data array, rather than a two-way matrix as in Figure 4.6(c). More conveniently, tripartite data can be represented by three

two-way matrices: for the Mische data, these were activist × organization, activist × event, and organization × event.

Multilevel networks

Recall that multilevel networks have two types of nodes (thought of as at different levels) with a bipartite network representing associations between the two types of nodes, and unipartite networks among the nodes at each level: that is, two types of nodes and three types of ties.

Figure 4.7 shows an example bipartite network and how it can be represented in three matrices. Here, following Wang et al. (2013), I have labelled the two types of nodes *A* - circular in the visualization in Figure 4.7(a)) – and *B* (square), so that we have an *A* network among the circular nodes, a *B* network among the square nodes and an *X* network of bipartite ties between *A* and *B* nodes. The *A*, *X* and *B* adjacency matrices are presented in Figure 4.7(b), (c) and (d), respectively. The



(a)

0	1	0
1	0	1
0	1	0

(b)

0	1	1	1
0	0	1	0
1	1	0	1

(c)

0	1	1	1
1	0	1	0
1	1	0	1
1	0	1	0

(d)

Figure 4.7 Multilevel network data structure

(a) visualization; (b) *A*-matrix (circular nodes); (c) *X*-matrix (bipartite); (d) *B*-matrix (square nodes)

A and *B* matrices are square with 0s down the diagonal, just as one would expect for unipartite whole networks. The X network is rectangular as for a bipartite network.

Of course, the three matrices in Figure 4.7 can be combined into one larger 7 × 7 matrix with the cells representing the links present in the visualization. In organizational and defence network research, Kathleen Carley (2003) proposed such a *meta-matrix* where the nodes represented people, resources and tasks: people have resources and tasks (as well as social ties), tasks require resources, and so on.

Longitudinal panel network data

The most commonly used network longitudinal design is for whole network data to be collected at multiple time-points. For instance, for the four-node network in Figure 4.1(a), the ties among the four nodes might be measured across three time periods. This assumes that the network ties change but the nodes are present throughout.

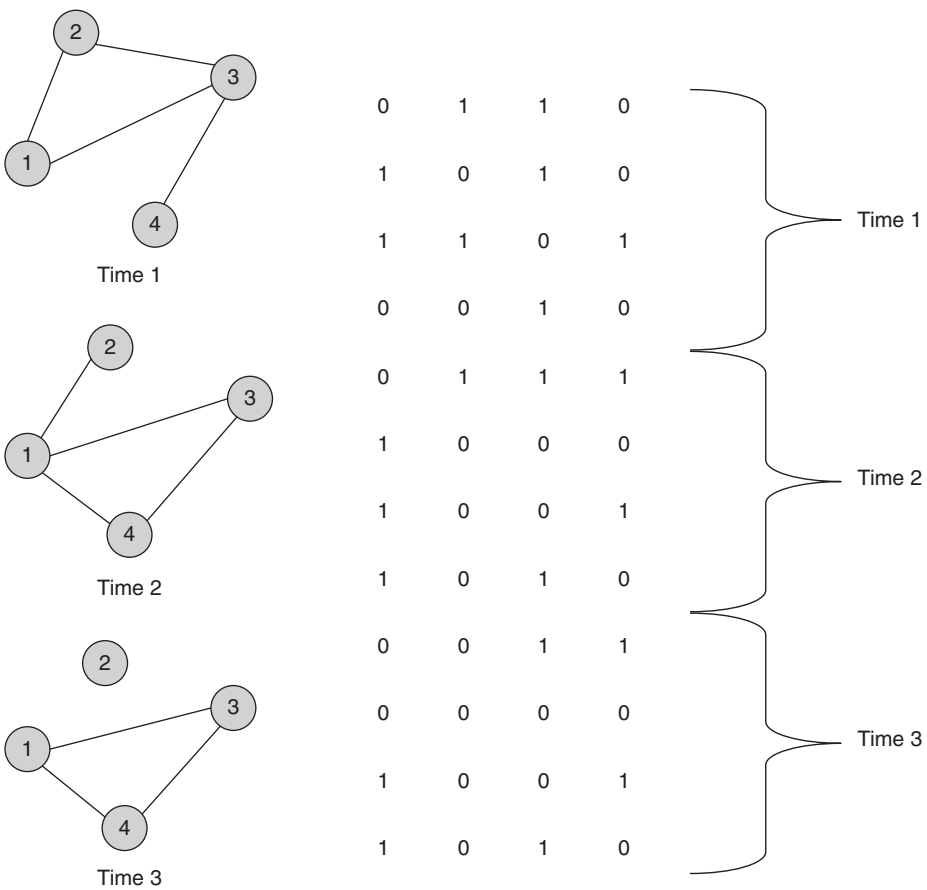


Figure 4.8 Panel longitudinal network data structure

This assumption may be appropriate in certain contexts, but not in others. In a school classroom, for example, the students may be unchanged across the one term, but their friendships may change.

In this case, we have three adjacency matrices, one for each time-point, which can be 'stacked' into the one rectangular matrix, as long as it is understood that the first *n* rows belongs to the first time-point, the second *n* to the second time-point, and so on (Figure 4.8). Alternatively, the three matrices can be stored in three separate data-files. In Figure 4.8, supposing that the relationship is friendship, we see that from time 1 to time 2, actor 2 ceases to be a friend with actor 3, and a new friendship is formed between actors 1 and 4. By time 3, actor 2 has ceased to be friends with the other actors.

Further complexity is introduced when the node set is not constant across time. For instance, a student may leave the school or a new student may arrive; or a company may cease to do business, or a new company may start up operations. So nodes can come into and out of existence. There is no unique method to enter network data with a changing node set: much depends on the proposed method of analysis. Sometimes researchers simply enter distinct panels for each time step, including both the adjacency matrix and a listing of the nodes present at that time. This would enable, for instance, calculation of the density at each time-point, irrespective of the nodes present. Sometimes it may be convenient to treat the boundary of the network as all nodes that appear at any time-point, but for a particular time the rows and columns for 'dead' nodes are forced to be zero.

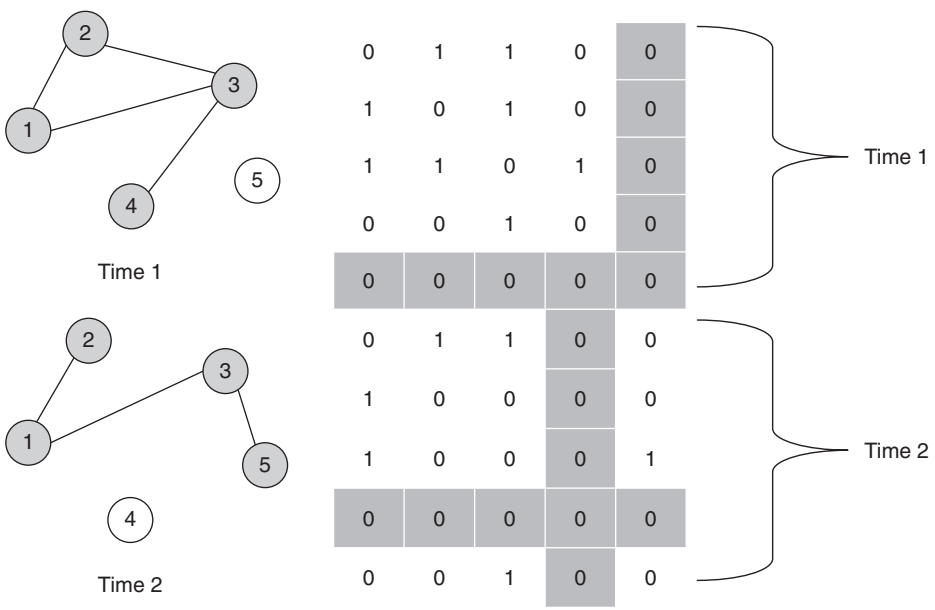


Figure 4.9 Panel network data with changing node set

For instance, in Figure 4.9, a two-panel longitudinal study, node 5 is not present at time 1 and node 4 is not present at time 2, as indicated by the non-filled nodes in the figure. There are some changes to ties among the nodes that are present across both time-points, and of course if a node is 'dead' at a particular time-point it can have no ties. Here the stacked matrix comprises two 5×5 adjacency matrices, but with row and column 5 fixed at 0 for time 1 and row and column 4 fixed at 0 for time 2 (these are shaded grey in the figure). The forced zeros need to be taken into account in doing any calculations: for instance, these cells need to be excluded when calculating the density, just as we excluded the diagonal cells from the original density calculation.

Event-based and other time-stamped relational data

Sometimes, instead of having panel network data, we have transactions between actors at a given time-point. For instance, the data might comprise telephone call data, where actor *i* phones actor *j* at time *t*. At its simplest, only the sequence of events might be recorded, perhaps in an edge list, where the first edge is the caller and receiver for the first phone call, the second edge the caller and the receiver for the second phone call, and so on. If the times that the calls started and ended are recorded, these would usually be entered as a third and fourth column, perhaps minutes or seconds after time 0, where time 0 could be the start of the first phone call.

Often event-based relational data need to be further processed for analysis. For instance, in some event history models, the number of possible phone calls at the given time (the 'risk set' of possible events at *t*) needs to be included in the dataset before analysis. This requires knowing which nodes are active at which time. Special purpose software is often used for this additional processing of the basic data structure.

Actor attributes: Putting individuals back into the network

Social networks are representations of social systems. Social systems are not just a collection of relational ties but also include individual actors. The best of social network research captures a balance between the individual and the system, between actor attributes and social structure. While much network research might focus on the network structure, it would be an unusual *social* network study that did not take into account individual variables of some type. Of course, the precise nature of the individual observations depends on the research question, the theoretical impetus behind the study, and the effects that need to be controlled in order to draw correct inferences about the processes at the centre of the research.

In this section, I describe the types of individual measures that can be used, the types of construct that might be considered, and appropriate data structures. I also make a few comments about dyadic covariates, which are network-like data structures that might help explain the network data under consideration. Often these are derived from observations on individuals. I will conclude the chapter by describing egonet data, a particular combination of individual and network variables.

Before we proceed, however, it is worth noting that sometimes research decisions need to be made about what counts as an individual, a node in our network. This is not a matter of defining the network boundary; rather, sometimes the definition of a node itself may be ambiguous. For most studies this issue is irrelevant and so is usually overlooked in texts on network analysis.

Let me explain. If the study is of students in a classroom, then the nodes are obviously the students. A study of organizations, however, is not always so clear-cut. Should a local government be seen as one organization, or should its different departments be treated as separate nodes? Should a bank be seen as one node, or separate bank branches? In a study of social ecological systems, Bodin and Tengö (2012) treated forests as separate nodes, but some forests were in close proximity, almost to the point of overlap. In this case the researchers could rely on distinctions between forests made by the local people, but in other contexts it may not be so clear-cut. When node definition is vague, there is no hard and fast solution, nor is there a general method. Rather, a careful research decision needs to be made based on theoretical grounds, knowledge of the specific context, or reasonable pragmatic considerations.

Types of individual constructs

The constructs to be considered are of course entirely dependent on the research question and include any individual effect relevant to the social system. In short, any construct used in other areas of social science is potentially applicable. Box 4.3 describes some of the types of individual constructs used in social network research, where the actors are people. Several different types may be relevant to the one study. There would be a different set of constructs for organizations, animals and so on.

These types of constructs will be familiar to social science researchers. It is at this point that other areas of social science most directly intersect with network research. If you have a carefully designed scale of attitudes, for instance, with reliability tested and the factor analytic structure of the scale well understood, this can still be used in social network research. Indeed, network researchers could learn much from the careful approach to observation of individual attributes adopted in other areas of social science. Too often, network researchers think that the skill lies in the network analysis, not in the observation and measurement. But with poor measurement of actor attributes, the finely tuned analysis is likely to prove meaningless in drawing conclusions about the individuals and possibly about the network structure.

BOX 4.3

Common types of individual constructs

1. *Demographic*: It is usual in social science research to collect some simple demographics on participants, such as sex or age. These remain relevant in social network research. For a start, it is good to report descriptive statistics about your actors, just as we would with a more traditional sample. But, additionally, the possibility of homophily effects for sex or age means that such attributes may be important factors in network structure.
2. *Social (or other) categories*: Individuals can belong to various groups that might be relevant in the network. Sometimes these might be considered demographic (e.g., ethnicity), but on other occasions the categories might be important aspects of the social setting within which the research takes place. For instance, in an organization, workers might be part of different divisions or work teams.
3. *Physical*: Certain physical attributes of the individual may be relevant, depending on the study. For instance, disease status is obviously relevant when studying the spread of contagious diseases. De la Haye et al. (2010) used the Body-Mass Index of adolescents in a study of eating behaviors in networks. External physical attributes may also be considered, such as the physical location of individuals if geospatial effects are considered.
4. *Behavioral*: The behavior of the actors can be relevant: for example, adoption of an innovation (Valente, 2005); a health-related behavior such as smoking or drinking (Light et al., 2013); teenage delinquency (Snijders and Baerveldt, 2003).
5. *Attitudinal*: The attitudes of individuals may be the topic of an influence study, where attitudinal change may be influenced by network partners. Attitudes may also affect other individual variables in a study, or the network structure itself.
6. *Psychological*: There is a small but growing body of work on the intersection of psychological factors, such as personality traits, and network structure (for more, see Chapter 6).

Types of individual measurement and data structures

There are basically three types of observations that can be applied to actor attributes. These are familiar from regular social science research.

- *Binary*: Binary observations include variables such as sex (male/female). It is often good to code these as 0 or 1.
- *Categorical*: Actors may be grouped into certain categories: for instance, ethnic group or work division. Numbers may be assigned to these categories but they are no more than indicators of similarities or differences (i.e. actors may be members of the same or different categorical group) and the actual values should not be considered.

  - Sometimes in social science, the categories may be ordered (*ordinal measurement*) so that a larger number does indicate a difference in a particular direction. Ranks are a good example: for instance, ordinal measures of a horse race indicate which horse comes 1st, 2nd, 3rd, …. Ordinal actor attributes are not as common in social network research.
- *Continuous*: Actors will have different ages, for instance, where the numbers represent a continuous measure of time.

Irrespective of the type of measure, each actor attribute variable can be entered into a dataset as a separate column, as is the standard practice in social science research. Mathematically, each attribute variable can be thought of as a *vector* (a single column of numbers). It is important that the row numbers coincide with the numbering of the nodes and hence with the row numbering of the adjacency matrix. In network visualizations, binary and categorical attributes are often represented with different colors on the nodes, and continuous attributes by different node sizes.

Figure 4.10 shows an example of the three different types of attribute variables for the network of four nodes in Figure 4.1. Here, the binary variable 'Sex' represents males (0) and females (1); the categorical variable 'Dept' represents three different departments in an organization; and the continuous variable 'Age' represents the age of the actors.

Node ID	Sex	Dept	Age
1	1	1	27
2	0	3	36
3	0	2	25
4	1	3	27

Figure 4.10 Attribute variables

For those who know some basic matrix algebra, simple results can be derived from a combination of the adjacency matrix and the attribute vector. For instance, multiplying the adjacency matrix by a binary attribute variable (coded 0 and 1) gives a vector with each row signifying how many network partners with attribute value 1 the row actor has. So, for instance, multiplying the adjacency matrix in Figure 4.1(c) by the vector for the attribute 'Sex' in Figure 4.10 gives the number of female network partners for each actor, as shown in Figure 4.11. We see, for instance, that actors 1 and 4 are female; actor 1 is a network partner of actors 2 and 3; so actor 1 has no female network partner. Actors 2 and 3 have one and two female partners, respectively, and actor 4 none. The entry for each row in the final vector is $\sum_{j=1}^{n} x_{ij} y_j$.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 0 \end{bmatrix}$$

Figure 4.11 Matrix multiplication of binary adjacency matrix and attribute vector

Matrix multiplication can also be used for other purposes in network analysis: for instance, multiplying an adjacency matrix by itself gives a matrix where each cell contains a count of the number of 2-paths between the two nodes. There are other different results, depending on the power and order of multiplication. These are the domain of network algebras. Mathematically inclined readers who are interested in following the topic further will find some directions in Chapter 9.

For most empirical network studies, however, we need not be too bothered by these details and you will typically not have to worry about matrix algebra.

Dyadic covariates

A study may include dyadic variables that, strictly speaking, do not constitute a social network, but do have the same data structure as a network. Such data are typically not at the centre of interest in a study but may be treated as 'covariates', in that the network structure that is the focus of attention may be affected in some way by the dyadic covariate. A good example is geospatial distance between actors. In some studies, the distance between actors (e.g., between the locations of their homes) may be theorized to affect the social network relationship between them. Current work on the integration of social networks in space is exemplified by articles in a recent special issue of the journal *Social Networks* (adams et al., 2012).

Dyadic covariates may also be derived from actor attribute data. For instance, if our interest is in age homophily, then the absolute difference in ages between pairs of actors can be treated as a dyadic covariate of the network tie. The age difference between each pair of actors is calculated, signs ignored, and the results placed in a dyadic covariate matrix as in Figure 4.12(a), which uses the age data from Figure 4.10 (mathematically, $|y_i - y_j|$). In this case, given that the data is an absolute difference, the matrix is naturally symmetric. In Figure 4.12(b), the categorical attribute variable 'Dept', describing an actor's department, is used to derive a binary dyadic covariate, 'Same department'. Figure 4.10 shows that only actors 2 and 4 come from the same department, so there is a 1 only in cells (2,4) and (4,2) in the matrix. Again the resulting matrix is naturally symmetric. A study involving such data could investigate

whether friendship was related to age difference or working in the same department, using these dyadic covariates.

Notice that a positive association between a network tie and age difference implies that a tie is more likely to occur when there is a larger difference in age: that is, in situations of heterophily (the opposite of homophily). So the direction of effect needs to be taken into account when interpreting the result.

0	9	2	0
9	0	11	9
2	11	0	2
0	9	2	0

(a)

0	0	0	0
0	0	0	1
0	0	0	0
0	1	0	0

(b)

Figure 4.12 Dyadic covariate matrices

(a) absolute difference in ages; (b) same department (derived from Figure 4.10)

Of course, there can be additional transformations applied in creating dyadic covariates. For instance, given that some spatial distances may be very large and some comparatively small, it may be helpful to use a logarithm of the distance between actors in analysis (Daraganova et al., 2012).

Egocentric network data

I have saved a description of egocentric network data structures until the end of this chapter, because although egocentric data are often the easiest to collect, several of the features discussed above are relevant.

Recall that an egonet comprises the network neighborhood around an actor (*ego*), including the network partners of *ego*, referred to as *alters*. The neighborhood includes the ties from ego to each alter, and often alter–alter ties. An egonet is akin to a one-wave snowball sample with a seed set size of one (ego). This data can be collected by survey, but may also be available by other means (electronically, or egos might be extracted from larger whole network data.)

There are several different levels of data here and it is often helpful to enter the data into three different files:

- An ego-level file which will contain all the attribute data collected on the *egos;*
- An alter-level file which will contain attribute data about each alter, as well as the ego to which each alter refers (this variable links cases across the different files), and data about the relationship between ego and alter (e.g., type of tie, strength of tie);
- For each ego, an alter–alter file containing data on the network ties between alters (if this data is collected).

Figure 4.13 illustrates the type of files from a small example with three egos (usually, of course, there would be many more). Here the ego-level file contains an ID for each ego and a variable for the sex of each (1 for female, 2 male). In a real study, of course, there may be many such variables.

The alter file contains ID variables for the alters as well as for the relevant ego. For instance, alter 11 is an alter for ego 1. In this case, I have coded the alter ID as a two-digit number with the first digit representing the relevant ego. Of course, this only works here because there are fewer than 10 egos and fewer than 10 alters per ego; more generally, some thought should be given to the most convenient method for numbering alter IDs. In an egonet study, it is typically assumed that the alters are different from one another (i.e. that the egonets do not overlap), so the coding is unique (e.g., alter 13 is not the same as alter 32). The ego-ID variable is important here because it links the alter and the ego files. In this particular example, there are three alters for each of the three egos, but of course that does not have to be so, and usually egos will differ in the number of alters.

Additionally, in the alter file we see a variable for sex of alter with the variable name 'Sex-A' to emphasize that this variable relates to alters, not to egos. There is also a variable named 'Close' for the closeness of the relationship, here measured as binary where '1' indicates a close relationship. Notice how the tie between ego and alter becomes an attribute variable for alter: this is a peculiarity of an egonet study.

Ego-ID	Sex
1	1
2	2
3	1

Ego file

Alter-ID	Ego-ID	Sex-A	Close
11	1	1	1
12	1	2	0
13	1	2	0
21	2	2	1
22	2	2	1
23	2	2	0
31	3	1	1
32	3	1	1
33	3	2	1

Alter file

0	1	1
1	0	0
1	0	0

Alter-alter file for Ego1

Figure 4.13 Data structure for an egonet study

Figure 4.13 also presents the alter–alter file for ego 1 in the form of an adjacency matrix. In this egonet, alter 11 is tied to alter 12 and 13 but there is no tie between alters 12 and 13. The egonet for ego 1 is visualized in Figure 4.14. The node for ego is a box, and nodes for alters are circles. A filled node indicates *Sex* and *Sex-A* = 1, and a thicker line represents *Close* = 1. Ego is of course tied to all alters but only to alter 11 with a thick line. Alters 12 and 13 are not tied.

There could be separate alter–alter files for each ego, but sometimes it is more convenient to enter this data into one file with an additional variable to indicate which matrix belongs to which ego. It is also possible to code this as an edge list, rather than a matrix. An edge list might include three columns, one for the ego and one each for the relevant pair of alters.

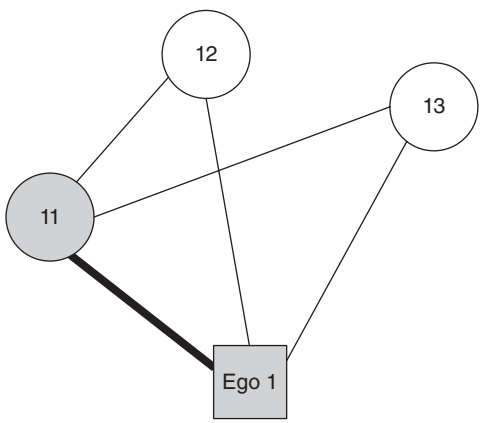


Figure 4.14 Egonet

For ego 1 from Figure 4.13

What is often done in terms of analysis is to create some additional variables from the alter–alter file to add to the ego file. For instance, the density of the alter–alter network indicates the extent of closure in ego's social environment: conversely, a lower density suggests that ego is in more of a brokerage position. Ignoring the variable *Close*, the alter–alter file for ego 1 is an undirected network and has two out of a possible three ties (remember ego and ego's ties are not included), so the density is 0.67. Let us suppose that the densities for the alter–alter networks for egos 2 and 3 (not shown in Figure 4.13) are 0.33 and 1.00, respectively.

In Figure 4.15, I have added these densities as a variable *Dens* to the ego file. Similarly, variables from the alter file can be aggregated into the ego file. I have also included additional variables: proportion of same sex alters (*Samesx*) and proportion of close partners (*Nclose*). For ego1, only one alter is the same sex and only one is close, so these proportions are 1/3.

Ego-ID	Sex	Dens	Samesx	Nclose
1	1	0.67	0.33	0.33
2	2	0.33	1.00	0.67
3	1	1.00	0.67	1.00

Figure 4.15 Expanded ego file

Including variables aggregated from the alter file and alter–alter files from Figure 4.13

There are usually important decisions to be made about the method of aggregation. In Figure 4.15, I could have used the number of same sex and close partners, rather than the proportion. Such choices are theoretical decisions that need to be thought through carefully. In this particular case, it makes no difference because each ego has three alters, but in reality the number of alters will differ and quite often dramatically. In that case, the counts of particular alters (e.g., females) may vary considerably from the proportion. Suppose, for instance, your egonet study is about social support. In order to feel well supported, is it sufficient to have a small number of strong friends who offer support, irrespective of how many friends you have, or is it necessary to have a high proportion of all your friends as supporters? The number and proportion have different theoretical implications.

Often the analysis will then be conducted using standard statistical techniques on the expanded ego file: for instance, in Figure 4.15 correlations and regressions could be conducted among the continuous variables and differences in means compared between sexes. Those readers familiar with hierarchical linear modelling (Snijders and Bosker, 2012) will note, however, that the datafile in Figure 4.15 and the alter file in Figure 4.13 are jointly suitable for multilevel modelling, with alters nested within egos. A multilevel model could then be used to predict alter attribute variables[1]. This could include predicting the type of relationship between ego and alter, given that in egonet studies the type of ego–alter tie (close or not close, in our example) becomes an attribute variable in the alter file.

In conclusion: The key point

As researchers, we all need to know how to enter data in the right form, but the right form itself reveals important aspects of the research design. So this is not just a *how*

[1]Do not confuse a *hierarchical linear model* – often called a 'multilevel model' – with multilevel networks. In this context, a multilevel model is a statistical model for nested data more generally, not specifically for networks. A multilevel network is a particular network data structure as explained earlier.

to do it chapter. Rather, an understanding of these different data structures will give an understanding of network-based research design.

The chapter also emphasizes – albeit, implicitly – *choice*. We have a variety of data structures. The first step is to decide which best applies to our study. This is not just a personal preference but crucially determines, and is determined by, our theoretical position. If we adopt bipartite data when the theoretical context demands a unipartite representation, we risk making horrible mistakes in inference. Once we have made the serious decision about why we should adopt a particular data structure, then we have to engage with the difficult question of which types of tie to investigate, which types of attributes and so on. These are not decisions based on numbers or measures, convenience or habit: these are matters of theory. I do not mean a 'Theory of X, Y or Z' but, rather, a theoretical argument which involves an understanding of the research context, previous work and existing knowledge. Then the decisions about the right data structures and the right variables can be made on the most solid ground.

BOX 4.4

Pulling back the curtain: What goes on in real network studies

For the sporting team study, we used a whole network approach. We had a complete list of current athletes in the club, so – as explained in Chapter 3 – the network boundary was relatively unambiguous. In Chapter 3, I described the difficult decisions made about the types of ties and attributes to measure. In the end we had several different types of ties, each of which were entered into a separate adjacency matrix analogous to that shown in Figure 4.3(c), with attribute variables entered into a datafile in the form shown in Figure 4.10.

For the organizational collaboration study, as explained in Chapter 3, the boundary was uncertain, so we employed a snowball sampling strategy in an effort to determine a sensible boundary. As in Figure 4.4, we had a core set of organizations that we knew were definitely relevant: these became our seed set. By asking informants from each of these organizations which other bodies were major collaboration partners of their own organization, we uncovered the collaboration ties among our seed set and snowballed out to a first zone of other organizations. We then sought to interview informants from those organizations, and so on. There is an issue of how many waves of snowball sampling are necessary. Ideally, there should be as many waves as required until no new organizations are named, but there are usually funding and timing constraints that need to be taken into account. For instance, if an organization is nominated by only one, or a small number, of the more central organizations, then perhaps that organization could be treated as peripheral and not interviewed. What counts as a 'small number' needs to be decided: there is no established rule.

In our case, the snowball approach after one wave seemed to capture the most central actors, with other actors more peripheral and of lower degree. Due to resource constraints, we concentrated our analysis on these central actors. At this point we reverted to a whole network study among these central actors, so did not explicitly invoke the data structures of Figures 4.4 and 4.5.

A question arises about whether we can find the entire boundary if the network is disconnected and our seed set only includes nodes that are likely to be tied to each other. Then it may be impossible to find separated components. In our case, due to local knowledge and the specific context, we were confident that major players would have some connection – perhaps indirect – to our seed set nodes, so this was not an issue for us. More generally, the extent of such a problem may be investigated by sampling some actors who seem more remote and disconnected from the original seed set, to see whether a snowball from them may eventually reconnect. The choice of such actors may not be obvious, however, and resource constraints in data collection need to be taken into account.