
Robert A. Hanneman and Mark Riddle

Introduction to social network methods

Table of contents

About this book

This on-line textbook introduces many of the basics of formal approaches to the analysis of social networks. The text relies heavily on the work of Freeman, Borgatti, and Everett (the authors of the UCINET software package). The materials here, and their organization, were also very strongly influenced by the text of Wasserman and Faust, and by a graduate seminar conducted by Professor Phillip Bonacich at UCLA. Many other users have also made very helpful comments and suggestions based on the first version. Errors and omissions, of course, are the responsibility of the authors.

You are invited to use and redistribute this text freely -- but please acknowledge the source.

Hanneman, Robert A. and Mark Riddle. 2005. Introduction to social network methods. Riverside, CA: University of California, Riverside (published in digital form at <http://faculty.ucr.edu/~hanneman/>)

Table of contents:

[Preface](#)

- [1. Social network data](#)
- [2. Why formal methods?](#)
- [3. Using graphs to represent social relations](#)
- [4. Working with Netdraw to visualize graphs](#)
- [5. Using matrices to represent social relations](#)
- [6. Working with network data](#)
- [7. Connection](#)
- [8. Embedding](#)
- [9. Ego networks](#)
- [10. Centrality and power](#)
- [11. Cliques and sub-groups](#)
- [12. Positions and roles: The idea of equivalence](#)
- [13. Measures of similarity and structural equivalence](#)
- [14. Automorphic equivalence](#)
- [15. Regular equivalence](#)
- [16. Multiplex networks](#)
- [17. Two-mode networks](#)
- [18. Some statistical tools](#)

[After word](#)

[Bibliography](#)

Introduction to social network methods

Preface

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

This book began as a set of reading notes as Hanneman sought to teach himself the basics of social network analysis. It then became a set of lecture notes for students in his undergraduate course in social network analysis. Through a couple extensions and revisions, it has evolved to cover more of the basic approaches to the analysis of social network data. Its current form, written in 2005, covers most of the algorithms and approaches that are collected in the computer package UCINET, version 6.85. Mark Riddle has added expertise in the statistical modeling of network data, study questions and problems, and connections to a variety of empirical literature that uses the techniques discussed here.

Our goal in preparing this book is to provide a very basic introduction to the core ideas of social network analysis, and how these ideas are implemented in the methodologies that many social network analysts use. The book is distributed free on the Internet in the hope that it may reach a diverse audience, and that the core ideas and methods of this field may be of interest. The book may also be suitable as course-support for undergraduate or introductory graduate training in social network analysis. While this text is not a user's guide to UCINET (which has excellent documentation in its help files), it may be of assistance to users working with that particular software package.

We hope that you will find things here that may stimulate your imagination. Social network analysis is a continuously and rapidly evolving field, and is one branch of the broader study of networks and complex systems. The concepts and techniques of social network analysis are informed by, and inform the evolution of these broader fields. We hope that this text will serve as a starting point.

[table of contents of the book](#)

Introduction to social network methods

1. Social network data

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

Contents of chapter 1: Social network data

- [Introduction: What's different about social network data?](#)
- [Nodes](#)
 - [Populations, samples, and boundaries](#)
 - [Modality and levels of analysis](#)
- [Relations](#)
 - [Sampling ties](#)
 - [Multiple relations](#)
- [Scales of measurement](#)
- [A note on statistics and social network data](#)

Introduction: What's different about social network data?

On one hand, there really isn't anything about social network data that is all that unusual. Social network analysts do use a specialized language for describing the structure and contents of the sets of observations that they use. But, network data can also be described and understood using the ideas and concepts of more familiar methods, like cross-sectional survey research.

On the other hand, the data sets that social network analysts develop usually end up looking quite different from the conventional rectangular data array so familiar to survey researchers and statistical analysts. The differences are quite important because they lead us to look at our data in a different way -- and even lead us to think differently about how to apply statistics.

"Conventional" social science data consist of a rectangular array of measurements. The rows of the array are the cases, or subjects, or observations. The columns consist of scores (quantitative or qualitative) on attributes, or variables, or measures. A simple example is shown as figure 1.1. Each cell of the array then describes the score of some actor (row) on some attribute (column). In some cases, there may be a third dimension to these arrays, representing panels of observations or multiple groups.

Figure 1.1. Example of rectangular data array

<i>Name</i>	<i>Sex</i>	<i>Age</i>	<i>In-Degree</i>
Bob	Male	32	2
Carol	Female	27	1
Ted	Male	29	1
Alice	Female	28	3

The fundamental data structure is one that leads us to compare how actors are similar or dissimilar to each other across attributes (by comparing rows). Or, perhaps more commonly, we examine how variables are

similar or dissimilar to each other in their distributions across actors (by comparing or correlating columns).

"Network" data (in their purest form) consist of a square array of measurements. The rows of the array are the cases, or subjects, or observations. The columns of the array are -- and note the key difference from conventional data -- the same set of cases, subjects, or observations. In each cell of the array describes a relationship between the actors. A simple example is shown as figure 1.2, which describes the network of friendship relations among four people.

Figure 1.2. Example of square array of network data

Who reports liking whom?				
	Choice:			
Chooser:	Bob	Carol	Ted	Alice
Bob	---	0	1	1
Carol	1	---	0	1
Ted	0	1	---	1
Alice	1	0	0	---

We could look at this data structure the same way as with attribute data. By comparing rows of the array, we can see which actors are similar to which other actors in whom they choose. By looking at the columns, we can see who is similar to whom in terms of being chosen by others. These are useful ways to look at the data, because they help us to see which actors have similar positions in the network. This is the first major emphasis of network analysis: seeing how actors are located or "embedded" in the overall network.

But a network analyst is also likely to look at the data structure in a second way -- holistically. The analyst might note that there are about equal numbers of ones and zeros in the matrix. This suggests that there is a moderate "density" of liking overall. The analyst might also compare the cells above and below the diagonal to see if there is reciprocity in choices (e.g. Bob chose Ted, did Ted choose Bob?). This is the second major emphasis of network analysis: seeing how the whole pattern of individual choices gives rise to more holistic patterns.

It is quite possible to think of the network data set in the same terms as "conventional data." One can think of the rows as simply a listing of cases, and the columns as attributes of each actor (i.e. the relations with other actors can be thought of as "attributes" of each actor). Indeed, many of the techniques used by network analysts (like calculating correlations and distances) are applied exactly the same way to network data as they would be to conventional data.

While it is possible to describe network data as just a special form of conventional data (and it is), network analysts look at the data in some rather fundamentally different ways. Rather than thinking about how an actor's ties with other actors describes the attributes of "ego," network analysts instead see a structure of connections, within which the actor is embedded. Actors are described by their relations, not by their attributes. And, the relations themselves are just as fundamental as the actors that they connect.

The major difference between conventional and network data is that conventional data focuses on actors and attributes; network data focus on actors and relations. The difference in emphasis is consequential for the choices that a researcher must make in deciding on research design, in conducting sampling, developing measurement, and handling the resulting data. It is not that the research tools used by network analysts are different from those of other social scientists (they mostly are not). But the special purposes and emphases of network research do call for some different considerations.

In this chapter, we will take a look at some of the issues that arise in design, sampling, and measurement for social network analysis. Our discussion will focus on the two parts of network data: nodes (or actors) and edges (or relations). We will try to show some of the ways in which network data are similar to, and different from more familiar actor by attribute data. We will introduce some new terminology that makes it easier to describe the special features of network data. Lastly, we will briefly discuss how the differences between network and actor-attribute data are consequential for the application of statistical tools.

[table of contents](#)

Nodes

Network data are defined by actors and by relations (or "nodes" and "edges"). The nodes or actors part of network data would seem to be pretty straight-forward. Other empirical approaches in the social sciences also think in terms of cases or subjects or sample elements and the like. There is one difference with most network data, however, that makes a big difference in how such data are usually collected -- and the kinds of samples and populations that are studied.

Network analysis focuses on the relations among actors, and not individual actors and their attributes. This means that the actors are usually not sampled independently, as in many other kinds of studies (most typically, surveys). Suppose we are studying friendship ties, for example. John has been selected to be in our sample. When we ask him, John identifies seven friends. We need to track down each of those seven friends and ask them about their friendship ties, as well. The seven friends are in our sample because John is (and vice-versa), so the "sample elements" are no longer "independent."

The nodes or actors included in non-network studies tend to be the result of independent probability sampling. Network studies are much more likely to include all of the actors who occur within some (usually naturally occurring) boundary. Often network studies don't use "samples" at all, at least in the conventional sense. Rather, they tend to include all of the actors in some population or populations. Of course, the populations included in a network study may be a sample of some larger set of populations. For example, when we study patterns of interaction among students in a classrooms, we include all of the children in a classroom (that is, we study the whole population of the classroom). The classroom itself, though, might have been selected by probability methods from a population of classrooms (say all of those in a school).

The use of whole populations as a way of selecting observations in (many) network studies makes it important for the analyst to be clear about the boundaries of each population to be studied, and how individual units of observation are to be selected within that population. Network data sets also frequently involve several levels of analysis, with actors embedded at the lowest level (i.e. network designs can be described using the language of "nested" designs).

[table of contents](#)

Populations, samples, and boundaries

Social network analysts rarely draw samples in their work. Most commonly, network analysts will identify some population and conduct a census (i.e. include all elements of the population as units of observation). A network analyst might examine all of the nouns and objects occurring in a text, all of the persons at a birthday party, all members of a kinship group, of an organization, neighborhood, or social class (e.g. landowners in a region, or royalty).

Survey research methods usually use a quite different approach to deciding which nodes to study. A list is made of all nodes (sometimes stratified or clustered), and individual elements are selected by probability methods. The logic of the method treats each individual as a separate "replication" that is, in a sense,

interchangeable with any other.

Because network methods focus on relations among actors, actors cannot be sampled independently to be included as observations. If one actor happens to be selected, then we must also include all other actors to whom our ego has (or could have) ties. As a result, network approaches tend to study whole populations by means of census, rather than by sample (we will discuss a number of exceptions to this shortly, under the topic of [sampling ties](#)).

The populations that network analysts study are remarkably diverse. At one extreme, they might consist of symbols in texts or sounds in verbalizations; at the other extreme, nations in the world system of states might constitute the population of nodes. Perhaps most common, of course, are populations of individual persons. In each case, however, the elements of the population to be studied are defined by falling within some boundary.

The boundaries of the populations studied by network analysts are of two main types. Probably most commonly, the boundaries are those imposed or created by the actors themselves. All the members of a classroom, organization, club, neighborhood, or community can constitute a population. These are naturally occurring clusters, or networks. So, in a sense, social network studies often draw the boundaries around a population that is known, *a priori*, to be a network. Alternatively, a network analyst might take a more "demographic" or "ecological" approach to defining population boundaries. We might draw observations by contacting all of the people who are found in a bounded spatial area, or who meet some criterion (having gross family incomes over \$1,000,000 per year). Here, we might have reason to suspect that networks exist, but the entity being studied is an abstract aggregation imposed by the investigator -- rather than a pattern of institutionalized social action that has been identified and labeled by its participants.

Network analysts can expand the boundaries of their studies by replicating populations. Rather than studying one neighborhood, we can study several. This type of design (which could use sampling methods to select populations) allows for replication and for testing of hypotheses by comparing populations. A second, and equally important way that network studies expand their scope is by the inclusion of multiple levels of analysis, or modalities.

[table of contents](#)

Modality and levels of analysis

The network analyst tends to see individual people nested within networks of face-to-face relations with other persons. Often these networks of interpersonal relations become "social facts" and take on a life of their own. A family, for example, is a network of close relations among a set of people. But this particular network has been institutionalized and given a name and reality beyond that of its component nodes. Individuals in their work relations may be seen as nested within organizations; in their leisure relations they may be nested in voluntary associations. Neighborhoods, communities, and even societies are, to varying degrees, social entities in and of themselves. And, as social entities, they may form ties with the individuals nested within them, and with other social entities.

Often network data sets describe the nodes and relations among nodes for a single bounded population. If I study the friendship patterns among students in a classroom, I am doing a study of this type. But a classroom exists within a school - which might be thought of as a network relating classes and other actors (principals, administrators, librarians, etc.). And most schools exist within school districts, which can be thought of as networks of schools and other actors (school boards, research wings, purchasing and personnel departments, etc.). There may even be patterns of ties among school districts (say by the exchange of students, teachers, curricular materials, etc.).

Most social network analysts think of individual persons as being embedded in networks that are embedded in networks that are embedded in networks. Network analysts describe such structures as "multi-modal." In our school example, individual students and teachers form one mode, classrooms a second, schools a third, and so on. A data set that contains information about two types of social entities (say persons and organizations) is a two mode network.

Of course, this kind of view of the nature of social structures is not unique to social network analystst. Statistical analysts deal with the same issues as "hierarchical" or "nested" designs. Theorists speak of the macro-meso-micro levels of analysis, or develop schema for identifying levels of analysis (individual, group, organization, community, institution, society, global order being perhaps the most commonly used system in sociology). One advantage of network thinking and method is that it naturally predisposes the analyst to focus on multiple levels of analysis simultaneously. That is, the network analyst is always interested in how the individual is embedded within a structure and how the structure emerges from the micro-relations between individual parts. The ability of network methods to map such multi-modal relations is, at least potentially, a step forward in rigor.

Having claimed that social network methods are particularly well suited for dealing with multiple levels of analysis and multi-modal data structures, it must immediately be admitted that social network analysis rarely actually takes much advantage. Most network analyses does move us beyond simple micro or macro reductionism -- and this is good. Few, if any, data sets and analyses, however, have attempted to work at more than two modes simultaneously. And, even when working with two modes, the most common strategy is to examine them more or less separately (one exception to this is the conjoint analysis of two mode networks). In chapter 17, we'll take a look at some methods for multi-mode networks.

[table of contents](#)

Relations

The other half of the design of network data has to do with what ties or relations are to be measured for the selected nodes. There are two main issues to be discussed here. In many network studies, all of the ties of a given type among all of the selected nodes are studied -- that is, a census is conducted. But, sometimes different approaches are used (because they are less expensive, or because of a need to generalize) that sample ties. There is also a second kind of sampling of ties that always occurs in network data. Any set of actors might be connected by many different kinds of ties and relations (e.g. students in a classroom might like or dislike each other, they might play together or not, they might share food or not, etc.). When we collect network data, we are usually selecting, or sampling, from among a set of kinds of relations that we might have measured.

[table of contents](#)

Sampling ties

Given a set of actors or nodes, there are several strategies for deciding how to go about collecting measurements on the relations among them. At one end of the spectrum of approaches are "full network" methods. This approach yields the maximum of information, but can also be costly and difficult to execute, and may be difficult to generalize. At the other end of the spectrum are methods that look quite like those used in conventional survey research. These approaches yield considerably less information about network structure, but are often less costly, and often allow easier generalization from the observations in the sample to some larger population. There is no one "right" method for all research questions and problems.

Full network methods require that we collect information about each actor's ties with all other actors. In

essence, this approach is taking a census of ties in a population of actors -- rather than a sample. For example we could collect data on shipments of copper between all pairs of nation states in the world system from International Monetary Fund records; we could examine the boards of directors of all public corporations for overlapping directors; we could count the number of vehicles moving between all pairs of cities; we could look at the flows of e-mail between all pairs of employees in a company; we could ask each child in a play group to identify their friends.

Because we collect information about ties between all pairs or dyads, full network data give a complete picture of relations in the population. Most of the special approaches and methods of network analysis that we will discuss in the remainder of this text were developed to be used with full network data. Full network data is necessary to properly define and measure many of the structural concepts of network analysis (e.g. between-ness).

Full network data allows for very powerful descriptions and analyses of social structures. Unfortunately, full network data can also be very expensive and difficult to collect. Obtaining data from every member of a population, and having every member rank or rate every other member can be very challenging tasks in any but the smallest groups. The task is made more manageable by asking respondents to identify a limited number of specific individuals with whom they have ties. These lists can then be compiled and cross-connected. But, for large groups (say all the people in a city), the task is practically impossible.

In many cases, the problems are not quite as severe as one might imagine. Most persons, groups, and organizations tend to have limited numbers of ties -- or at least limited numbers of strong ties. This is probably because social actors have limited resources, energy, time, and cognitive capacity -- and cannot maintain large numbers of strong ties. It is also true that social structures can develop a considerable degree of order and solidarity with relatively few connections.

Snowball methods begin with a focal actor or set of actors. Each of these actors is asked to name some or all of their ties to other actors. Then, all the actors named (who were not part of the original list) are tracked down and asked for some or all of their ties. The process continues until no new actors are identified, or until we decide to stop (usually for reasons of time and resources, or because the new actors being named are very marginal to the group we are trying to study).

The snowball method can be particularly helpful for tracking down "special" populations (often numerically small sub-sets of people mixed in with large numbers of others). Business contact networks, community elites, deviant sub-cultures, avid stamp collectors, kinship networks, and many other structures can be pretty effectively located and described by snowball methods. It is sometimes not as difficult to achieve closure in snowball "samples" as one might think. The limitations on the numbers of strong ties that most actors have, and the tendency for ties to be reciprocated often make it fairly easy to find the boundaries.

There are two major potential limitations and weaknesses of snowball methods. First, actors who are not connected (i.e. "isolates") are not located by this method. The presence and numbers of isolates can be a very important feature of populations for some analytic purposes. The snowball method may tend to overstate the "connectedness" and "solidarity" of populations of actors. Second, there is no guaranteed way of finding all of the connected individuals in the population. Where does one start the snowball rolling? If we start in the wrong place or places, we may miss whole sub-sets of actors who are connected -- but not attached to our starting points.

Snowball approaches can be strengthened by giving some thought to how to select the initial nodes. In many studies, there may be a natural starting point. In community power studies, for example, it is common to begin snowball searches with the chief executives of large economic, cultural, and political organizations. While such an approach will miss most of the community (those who are "isolated" from the elite network), the approach is very likely to capture the elite network quite effectively.

Ego-centric networks (with alter connections)

In many cases it will not be possible (or necessary) to track down the full networks beginning with focal nodes (as in the snowball method). An alternative approach is to begin with a selection of focal nodes (egos), and identify the nodes to which they are connected. Then, we determine which of the nodes identified in the first stage are connected to one another. This can be done by contacting each of the nodes; sometimes we can ask ego to report which of the nodes that it is tied to are tied to one another.

This kind of approach can be quite effective for collecting a form of relational data from very large populations, and can be combined with attribute-based approaches. For example, we might take a simple random sample of male college students and ask them to report who are their close friends, and which of these friends know one another. This kind of approach can give us a good and reliable picture of the kinds of networks (or at least the local neighborhoods) in which individuals are embedded. We can find out such things as how many connections nodes have, and the extent to which these nodes are close-knit groups. Such data can be very useful in helping to understand the opportunities and constraints that ego has as a result of the way they are embedded in their networks.

The ego-centered approach with alter connections can also give us some information about the network as a whole, though not as much as snowball or census approaches. Such data are, in fact, micro-network data sets -- samplings of local areas of larger networks. Many network properties -- distance, centrality, and various kinds of positional equivalence cannot be assessed with ego-centric data. Some properties, such as overall network density can be reasonably estimated with ego-centric data. Some properties -- such as the prevalence of reciprocal ties, cliques, and the like can be estimated rather directly.

Ego-centric networks (ego only)

Ego-centric methods really focus on the individual, rather than on the network as a whole. By collecting information on the connections among the actors connected to each focal ego, we can still get a pretty good picture of the "local" networks or "neighborhoods" of individuals. Such information is useful for understanding how networks affect individuals, and they also give a (incomplete) picture of the general texture of the network as a whole.

Suppose, however, that we only obtained information on ego's connections to alters -- but not information on the connections among those alters. Data like these are not really "network" data at all. That is, they cannot be represented as a square actor-by-actor array of ties. But doesn't mean that ego-centric data without connections among the alters are of no value for analysts seeking to take a structural or network approach to understanding actors. We can know, for example, that some actors have many close friends and kin, and others have few. Knowing this, we are able to understand something about the differences in the actors places in social structure, and make some predictions about how these locations constrain their behavior. What we cannot know from ego-centric data with any certainty is the nature of the macro-structure or the whole network.

In ego-centric networks, the alters identified as connected to each ego are probably a set that is unconnected with those for each other ego. While we cannot assess the overall density or connectedness of the population, we can sometimes be a bit more general. If we have some good theoretical reason to think about alters in terms of their social roles, rather than as individual occupants of social roles, ego-centered networks can tell us a good bit about local social structures. For example, if we identify each of the alters connected to an ego by a friendship relation as "kin," "co-worker," "member of the same church," etc., we can build up a picture of the networks of social positions (rather than the networks of individuals) in which egos are embedded. Such an approach, of course, assumes that such categories as "kin" are real and meaningful determinants of patterns of interaction.

[table of contents](#)

Multiple relations

In a conventional actor-by-trait data set, each actor is described by many variables (and each variable is realized in many actors). In the most common social network data set of actor-by-actor ties, only one kind of relation is described. Just as we often are interested in multiple attributes of actors, we are often interested in multiple kinds of ties that connect actors in a network.

In thinking about the network ties among faculty in an academic department, for example, we might be interested in which faculty have students in common, serve on the same committees, interact as friends outside of the workplace, have one or more areas of expertise in common, and co-author papers. The positions that actors hold in the web of group affiliations are multi-faceted. Positions in one set of relations may re-enforce or contradict positions in another (I might share friendship ties with one set of people with whom I do not work on committees, for example). Actors may be tied together closely in one relational network, but be quite distant from one another in a different relational network. The locations of actors in multi-relational networks and the structure of networks composed of multiple relations are some of the most interesting (and still relatively unexplored) areas of social network analysis.

When we collect social network data about certain kinds of relations among actors we are, in a sense, sampling from a population of possible relations. Usually our research question and theory indicate which of the kinds of relations among actors are the most relevant to our study, and we do not sample -- but rather select -- relations. In a study concerned with economic dependency and growth, for example, I could collect data on the exchange of performances by musicians between nations -- but it is not really likely to be all that relevant.

If we do not know what relations to examine, how might we decide? There are a number of conceptual approaches that might be of assistance. Systems theory, for example, suggests two domains: material and informational. Material things are "conserved" in the sense that they can only be located at one node of the network at a time. Movements of people between organizations, money between people, automobiles between cities, and the like are all examples of material things which move between nodes -- and hence establish a network of material relations. Informational things, to the systems theorist, are "non-conserved" in the sense that they can be in more than one place at the same time. If I know something and share it with you, we both now know it. In a sense, the commonality that is shared by the exchange of information may also be said to establish a tie between two nodes. One needs to be cautious here, however, not to confuse the simple possession of a common attribute (e.g. gender) with the presence of a tie (e.g. the exchange of views between two persons on issues of gender).

Methodologies for working with multi-relational data are not as well developed as those for working with single relations. Many interesting areas of work such as network correlation, multi-dimensional scaling and clustering, and role algebras have been developed to work with multi-relational data. For the most part, these topics are beyond the scope of the current text, and are best approached after the basics of working with single relational networks are mastered. We will look at some methods for multi-relational (a.k.a. "multiplex" network data in chapter 16).

[table of contents](#)

Scales of measurement

Like other kinds of data, the information we collect about ties between actors can be measured (i.e. we can assign scores to our observations) at different "levels of measurement." The different levels of measurement

are important because they limit the kinds of questions that can be examined by the researcher. Scales of measurement are also important because different kinds of scales have different mathematical properties, and call for different algorithms in describing patterns and testing inferences about them.

It is conventional to distinguish nominal, ordinal, and interval levels of measurement (the ratio level can, for all practical purposes, be grouped with interval). It is useful, however, to further divide nominal measurement into binary and multi-category variations; it is also useful to distinguish between full-rank ordinal measures and grouped ordinal measures. We will briefly describe all of these variations, and provide examples of how they are commonly applied in social network studies.

Binary measures of relations: By far the most common approach to scaling (assigning numbers to) relations is to simply distinguish between relations being absent (coded zero), and ties being present (coded one). If we ask respondents in a survey to tell us "which other people on this list do you like?" we are doing binary measurement. Each person from the list that is selected is coded one. Those who are not selected are coded zero.

Much of the development of graph theory in mathematics, and many of the algorithms for measuring properties of actors and networks have been developed for binary data. Binary data is so widely used in network analysis that it is not unusual to see data that are measured at a "higher" level transformed into binary scores before analysis proceeds. To do this, one simply selects some "cut point" and re-scores cases as below the cut-point (zero) or above it (one). Dichotomizing data in this way is throwing away information. The analyst needs to consider what is relevant (i.e. what is the theory about? is it about the presence and pattern of ties, or about the strengths of ties?), and what algorithms are to be applied in deciding whether it is reasonable to recode the data. Very often, the additional power and simplicity of analysis of binary data is "worth" the cost in information lost.

Multiple-category nominal measures of relations: In collecting data we might ask our respondents to look at a list of other people and tell us: "for each person on this list, select the category that describes your relationship with them the best: friend, lover, business relationship, kin, or no relationship." We might score each person on the list as having a relationship of type "1" type "2" etc. This kind of a scale is nominal or qualitative -- each person's relationship to the subject is coded by its type, rather than its strength. Unlike the binary nominal (true-false) data, the multiple category nominal measure is multiple choice.

The most common approach to analyzing multiple-category nominal measures is to use it to create a series of binary measures. That is, we might take the data arising from the question described above and create separate sets of scores for friendship ties, for lover ties, for kin ties, etc. This is very similar to "dummy coding" as a way of handling multiple choice types of measures in statistical analysis. In examining the resulting data, however, one must remember that each node was allowed to have a tie in at most one of the resulting networks. That is, a person can be a friendship tie or a lover tie -- but not both -- as a result of the way we asked the question. In examining the resulting networks, densities may be artificially low, and there will be an inherent negative correlation among the matrices.

This sort of multiple choice data can also be "binarized." That is, we can ignore what kind of tie is reported, and simply code whether a tie exists for a dyad, or not. This may be fine for some analyses -- but it does waste information. One might also wish to regard the types of ties as reflecting some underlying continuous dimension (for example, emotional intensity). The types of ties can then be scaled into a single grouped ordinal measure of tie strength. The scaling, of course, reflects the predispositions of the analyst -- not the reports of the respondents.

Grouped ordinal measures of relations: One of the earliest traditions in the study of social networks asked respondents to rate each of a set of others as "liked" "disliked" or "neutral." The result is a grouped ordinal scale (i.e., there can be more than one "liked" person, and the categories reflect an underlying rank order of

intensity). Usually, this kind of three point scale was coded -1, 0, and +1 to reflect negative liking, indifference, and positive liking. When scored this way, the pluses and minuses make it fairly easy to write algorithms that will count and describe various network properties (e.g. the structural balance of the graph).

Grouped ordinal measures can be used to reflect a number of different quantitative aspects of relations. Network analysts are often concerned with describing the "strength" of ties. But, "strength" may mean (some or all of) a variety of things. One dimension is the frequency of interaction -- do actors have contact daily, weekly, monthly, etc. Another dimension is "intensity," which usually reflects the degree of emotional arousal associated with the relationship (e.g. kin ties may be infrequent, but carry a high "emotional charge" because of the highly ritualized and institutionalized expectations). Ties may be said to be stronger if they involve many different contexts or types of ties. Summing nominal data about the presence or absence of multiple types of ties gives rise to an ordinal (actually, interval) scale of one dimension of tie strength. Ties are also said to be stronger to the extent that they are reciprocated. Normally we would assess reciprocity by asking each actor in a dyad to report their feelings about the other. However, one might also ask each actor for their perceptions of the degree of reciprocity in a relation: Would you say that neither of you like each other very much, that you like X more than X likes you, that X likes you more than you like X, or that you both like each other about equally?

Ordinal scales of measurement contain more information than nominal. That is, the scores reflect finer gradations of tie strength than the simple binary "presence or absence." This would seem to be a good thing, yet it is frequently difficult to take advantage of ordinal data. The most commonly used algorithms for the analysis of social networks have been designed for binary data. Many have been adapted to continuous data -- but for interval, rather than ordinal scales of measurement. Ordinal data, consequently, are often binarized by choosing some cut-point and re-scoring. Alternatively, ordinal data are sometimes treated as though they really were interval. The former strategy has some risks, in that choices of cut-points can be consequential; the latter strategy has some risks, in that the intervals separating points on an ordinal scale may be very heterogeneous.

Full-rank ordinal measures of relations: Sometimes it is possible to score the strength of all of the relations of an actor in a rank order from strongest to weakest. For example, I could ask each respondent to write a "1" next to the name of the person in the class that you like the most, a "2" next to the name of the person you like next most, etc. The kind of scale that would result from this would be a "full rank order scale." Such scales reflect differences in degree of intensity, but not necessarily equal differences -- that is, the difference between my first and second choices is not necessarily the same as the difference between my second and third choices. Each relation, however, has a unique score (1st, 2nd, 3rd, etc.).

Full rank ordinal measures are somewhat uncommon in the social networks research literature, as they are in most other traditions. Consequently, there are relatively few methods, definitions, and algorithms that take specific and full advantage of the information in such scales. Most commonly, full rank ordinal measures are treated as if they were interval. There is probably somewhat less risk in treating fully rank ordered measures (compared to grouped ordinal measures) as though they were interval, though the assumption is still a risky one. Of course, it is also possible to group the rank order scores into groups (i.e. produce a grouped ordinal scale) or dichotomize the data (e.g. the top three choices might be treated as ties, the remainder as non-ties). In combining information on multiple types of ties, it is frequently necessary to simplify full rank order scales. But, if we have a number of full rank order scales that we may wish to combine to form a scale (i.e. rankings of people's likings of other in the group, frequency of interaction, etc.), the sum of such scales into an index is plausibly treated as a truly interval measure.

Interval measures of relations: The most "advanced" level of measurement allows us to discriminate among the relations reported in ways that allow us to validly state that, for example, "this tie is twice as strong as that tie." Ties are rated on scales in which the difference between a "1" and a "2" reflects the same

amount of real difference as that between "23" and "24."

True interval level measures of the strength of many kinds of relationships are fairly easy to construct, with a little imagination and persistence. Asking respondents to report the details of the frequency or intensity of ties by survey or interview methods, however, can be rather unreliable -- particularly if the relationships being tracked are not highly salient and infrequent. Rather than asking whether two people communicate, one could count the number of email, phone, and inter-office mail deliveries between them. Rather than asking whether two nations trade with one another, look at statistics on balances of payments. In many cases, it is possible to construct interval level measures of relationship strength by using artifacts (e.g. statistics collected for other purposes) or observation.

Continuous measures of the strengths of relationships allow the application of a wider range of mathematical and statistical tools to the exploration and analysis of the data. Many of the algorithms that have been developed by social network analysts, originally for binary data, have been extended to take advantage of the information available in full interval measures. Whenever possible, connections should be measured at the interval level -- as we can always move to a less refined approach later; if data are collected at the nominal level, it is much more difficult to move to a more refined level.

Even though it is a good idea to measure relationship intensity at the most refined level possible, most network analysis does not operate at this level. The most powerful insights of network analysis, and many of the mathematical and graphical tools used by network analysts were developed for simple graphs (i.e. binary, undirected). Many characterizations of the embeddedness of actors in their networks, and of the networks themselves are most commonly thought of in discrete terms in the research literature. As a result, it is often desirable to reduce even interval data to the binary level by choosing a cutting -point, and coding tie strength above that point as "1" and below that point as "0." Unfortunately, there is no single "correct" way to choose a cut-point. Theory and the purposes of the analysis provide the best guidance. Sometimes examining the data can help (maybe the distribution of tie strengths really is discretely bi-modal, and displays a clear cut point; maybe the distribution is highly skewed and the main feature is a distinction between no tie and any tie). When a cut-point is chosen, it is wise to also consider alternative values that are somewhat higher and lower, and repeat the analyses with different cut-points to see if the substance of the results is affected. This can be very tedious, but it is very necessary. Otherwise, one may be fooled into thinking that a real pattern has been found, when we have only observed the consequences of where we decided to put our cut-point.

[table of contents](#)

A note on statistics and social network data

Social network analysis is more a branch of "mathematical" sociology than of "statistical or quantitative analysis," though social network analysts most certainly practice both approaches. The distinction between the two approaches is not clear-cut. Mathematical approaches to network analysis tend to treat the data as "deterministic." That is, they tend to regard the measured relationships and relationship strengths as accurately reflecting the "real" or "final" or "equilibrium" status of the network. Mathematical types also tend to assume that the observations are not a "sample" of some larger population of possible observations; rather, the observations are usually regarded as the population of interest. Statistical analysts tend to regard the particular scores on relationship strengths as stochastic or probabilistic realizations of an underlying true tendency or probability distribution of relationship strengths. Statistical analysts also tend to think of a particular set of network data as a "sample" of a larger class or population of such networks or network elements -- and have a concern for the results of the current study would be reproduced in the "next" study of similar samples.

In the chapters that follow in this text, we will mostly be concerned with the "mathematical" rather than the "statistical" side of network analysis (again, it is important to remember that I am over-drawing the

differences in this discussion). Before passing on to this, we should note a couple main points about the relationship between the material that you will be studying here, and the main statistical approaches in sociology. In chapter 18, we will explore some of the basic ways in which statistical tools have been adapted to study social network data.

In one way, there is little apparent difference between conventional statistical approaches and network approaches. Univariate, bi-variate, and even many multivariate descriptive statistical tools are commonly used in the describing, exploring, and modeling social network data. Social network data are, as we have pointed out, easily represented as arrays of numbers -- just like other types of sociological data. As a result, the same kinds of operations can be performed on network data as on other types of data. Algorithms from statistics are commonly used to describe characteristics of individual observations (e.g. the median tie strength of actor X with all other actors in the network) and the network as a whole (e.g. the mean of all tie strengths among all actors in the network). Statistical algorithms are very heavily used in assessing the degree of similarity among actors, and in finding patterns in network data (e.g. factor analysis, cluster analysis, multi-dimensional scaling). Even the tools of predictive modeling are commonly applied to network data (e.g. correlation and regression).

Descriptive statistical tools are really just algorithms for summarizing characteristics of the distributions of scores. That is, they are mathematical operations. Where statistics really become "statistical" is on the inferential side. That is, when our attention turns to assessing the reproducibility or likelihood of the pattern that we have described. Inferential statistics can be, and are, applied to the analysis of network data. But, there are some quite important differences between the flavors of inferential statistics used with network data, and those that are most commonly taught in basic courses in statistical analysis in sociology.

Probably the most common emphasis in the application of inferential statistics to social science data is to answer questions about the stability, reproducibility, or generalizability of results observed in a single sample. The main question is: if I repeated the study on a different sample (drawn by the same method), how likely is it that I would get the same answer about what is going on in the whole population from which I drew both samples? This is a really important question -- because it helps us to assess the confidence (or lack of it) that we ought to have in assessing our theories and giving advice.

To the extent the observations used in a network analysis are drawn by probability sampling methods from some identifiable population of actors and/or ties, the same kind of question about the generalizability of sample results applies. Often this type of inferential question is of little interest to social network researchers. In many cases, they are studying a particular network or set of networks, and have no interest in generalizing to a larger population of such networks (either because there isn't any such population, or we don't care about generalizing to it in any probabilistic way). In some other cases we may have an interest in generalizing, but our sample was not drawn by probability methods. Network analysis often relies on artifacts, direct observation, laboratory experiments, and documents as data sources -- and usually there are no plausible ways of identifying populations and drawing samples by probability methods.

The other major use of inferential statistics in the social sciences is for testing hypotheses. In many cases, the same or closely related tools are used for questions of assessing generalizability and for hypothesis testing. The basic logic of hypothesis testing is to compare an observed result in a sample to some null hypothesis value, relative to the sampling variability of the result under the assumption that the null hypothesis is true. If the sample result differs greatly from what was likely to have been observed under the assumption that the null hypothesis is true -- then the null hypothesis is probably not true.

The key link in the inferential chain of hypothesis testing is the estimation of the standard errors of statistics. That is, estimating the expected amount that the value of a statistic would "jump around" from one sample to the next simply as a result of accidents of sampling. We rarely, of course, can directly observe or calculate such standard errors -- because we don't have replications. Instead, information from our sample is used to

estimate the sampling variability.

With many common statistical procedures, it is possible to estimate standard errors by well validated approximations (e.g. the standard error of a mean is usually estimated by the sample standard deviation divided by the square root of the sample size). These approximations, however, hold when the observations are drawn by independent random sampling. Network observations are almost always non-independent, by definition. Consequently, conventional inferential formulas do not apply to network data (though formulas developed for other types of dependent sampling may apply). It is particularly dangerous to assume that such formulas do apply, because the non-independence of network observations will usually result in under-estimates of true sampling variability -- and hence, too much confidence in our results.

The approach of most network analysts interested in statistical inference for testing hypotheses about network properties is to work out the probability distributions for statistics directly. This approach is used because: 1) no one has developed approximations for the sampling distributions of most of the descriptive statistics used by network analysts and 2) interest often focuses on the probability of a parameter relative to some theoretical baseline (usually randomness) rather than on the probability that a given network is typical of the population of all networks.

Suppose, for example, that I was interested in the proportion of the actors in a network who were members of cliques (or any other network statistic or parameter). The notion of a clique implies structure -- non-random connections among actors. I have data on a network of ten nodes, in which there are 20 symmetric ties among actors, and I observe that there is one clique containing four actors. The inferential question might be posed as: how likely is it, if ties among actors were purely random events, that a network composed of ten nodes and 20 symmetric ties would display one or more cliques of size four or more? If it turns out that cliques of size four or more in random networks of this size and degree are quite common, I should be very cautious in concluding that I have discovered "structure" or non-randomness. If it turns out that such cliques (or more numerous or more inclusive ones) are very unlikely under the assumption that ties are purely random, then it is very plausible to reach the conclusion that there is a social structure present.

But how can I determine this probability? The method used is one of simulation -- and, like most simulation, a lot of computer resources and some programming skills are often necessary. In the current case, I might use a table of random numbers to distribute 20 ties among 10 actors, and then search the resulting network for cliques of size four or more. If no clique is found, I record a zero for the trial; if a clique is found, I record a one. The rest is simple. Just repeat the experiment several thousand times and add up what proportion of the "trials" result in "successes." The probability of a success across these simulation experiments is a good estimator of the likelihood that I might find a network of this size and density to have a clique of this size "just by accident" when the non-random causal mechanisms that I think cause cliques are not, in fact, operating.

This may sound odd, and it is certainly a lot of work (most of which, thankfully, can be done by computers). But, in fact, it is not really different from the logic of testing hypotheses with non-network data. Social network data tend to differ from more "conventional" survey data in some key ways: network data are often not probability samples, and the observations of individual nodes are not independent. These differences are quite consequential for both the questions of generalization of findings, and for the mechanics of hypothesis testing. There is, however, nothing fundamentally different about the logic of the use of descriptive and inferential statistics with social network data.

The application of statistics to social network data is an interesting area, and one that is, at the time of this writing, at a "cutting edge" of research in the area. Since this text focuses on more basic and commonplace uses of network analysis, we won't have very much more to say about statistics beyond this point. You can think of much of what follows here as dealing with the "descriptive" side of statistics (developing index numbers to describe certain aspects of the distribution of relational ties among actors in networks). For those with an interest in the inferential side, a good place to start is with the second half of the excellent

Wasserman and Faust textbook.

[table of contents](#)

[table of contents of the book](#)

Introduction to Social Network Methods

2. Why formal methods?

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

Contents of chapter 2: Why formal methods?

- [Introduction](#)
 - [Efficiency](#)
 - [Using computers](#)
 - [Seeing patterns](#)
 - [Summary](#)
-

Introduction:

The basic idea of a social network is very simple. A social network is a set of actors (or points, or nodes, or agents) that may have relationships (or edges, or ties) with one another. Networks can have few or many actors, and one or more kinds of relations between pairs of actors. To build a useful understanding of a social network, a complete and rigorous description of a pattern of social relationships is a necessary starting point for analysis. That is, ideally we will know about all of the relationships between each pair of actors in the population.

The amount of information that we need to describe even small social networks can be quite great. Managing these data, and manipulating them so that we can see patterns of social structure can be tedious and complicated. All of the tasks of social network methods are made easier by using tools from mathematics. For the manipulation of network data, and the calculation of indexes describing networks, it is most useful to record information as matrices. For visualizing patterns, graphs are often useful.

Efficiency

One reason for using mathematical and graphical techniques in social network analysis is to represent the descriptions of networks compactly and systematically. This also enables us to use computers to store and manipulate the information quickly and more accurately than we can by hand. For small populations of actors (e.g. the people in a neighborhood, or the business firms in an industry), we can describe the pattern of social relationships that connect the actors rather completely and effectively using words. To make sure that our description is complete, however, we might want to list all logically possible pairs of actors, and describe each kind of possible relationship for each pair. This can get pretty tedious if the number of actors and/or number of kinds of relations is large. Formal representations ensure that all the necessary information is systematically represented, and provides rules for doing so in ways that are much more efficient than lists.

Using computers

A related reason for using (particularly mathematical) formal methods for representing social networks is that mathematical representations allow us to apply computers to the analysis of network data. Why this is important will become clearer as we learn more about how structural analysis of social networks occurs.

Suppose, for a simple example, we had information about trade-flows of 50 different commodities (e.g. coffee, sugar, tea, copper, bauxite) among the 170 or so nations of the world system in a given year. Here, the 170 nations can be thought of as actors or nodes, and the amount of each commodity exported from each nation to each of the other 169 can be thought of as the strength of a directed tie from the focal nation to the other. A social scientist might be interested in whether the "structures" of trade in mineral products are more similar to one another than, the structure of trade in mineral products are to vegetable products. To answer this fairly simple (but also pretty important) question, a huge amount of manipulation of the data is necessary. It could take, literally, years to do by hand; it can be done by a computer in a few minutes.

Seeing patterns

The third, and final reason for using "formal" methods (mathematics and graphs) for representing social network data is that the techniques of graphing and the rules of mathematics themselves suggest things that we might look for in our data — things that might not have occurred to us if we presented our data using descriptions in words. Again, allow me a simple example.

Suppose we were describing the structure of close friendship in a group of four people: Bob, Carol, Ted, and Alice. This is easy enough to do with words. Suppose that Bob likes Carol and Ted, but not Alice; Carol likes Ted, but neither Bob nor Alice; Ted likes all three of the other members of the group; and Alice likes only Ted (this description should probably strike you as being a description of a very unusual social structure).

We could also describe this pattern of liking ties with an actor-by-actor matrix where the rows represent choices by each actor. We will put in a "1" if an actor likes another, and a "0" if they don't. Such a matrix would look like figure 2.1.

Figure 2.1. Matrix representation of "liking" relation among four actors

	Bob	Carol	Ted	Alice
Bob	---	1	1	0
Carol	0	---	1	0
Ted	1	1	---	1
Alice	0	0	1	---

There are lots of things that might immediately occur to us when we see our data arrayed in this way, that we might not have thought of from reading the description of the pattern of ties in words. For example, our eye is led to scan across each row; we notice that Ted likes more people than Bob, than Alice and Carol. Is it possible that there is a pattern here? Are men are more likely to report ties of liking than women are (actually, research literature suggests that this is not generally true). Using a "matrix representation" also immediately raises a question: the locations on the main diagonal (e.g. Bob likes Bob, Carol likes Carol) are empty. Is this a reasonable thing? Or, should our description of the pattern of liking in the group include some statements about "self-liking"? There isn't any right answer to this question. My point is just that using a matrix to represent the pattern of ties among actors may let us see some patterns more easily, and may cause us to ask some questions (and maybe even some useful ones) that a verbal description doesn't stimulate.

Summary

There are three main reasons for using "formal" methods in representing social network data:

- Matrices and graphs are compact and systematic: They summarize and present a lot of information quickly and easily; and they force us to be systematic and complete in describing patterns of social relations.
- Matrices and graphs allow us to apply computers to analyzing data: This is helpful because doing systematic analysis of social network data can be extremely tedious if the number of actors or number of types of relationships among the actors is large. Most of the work is dull, repetitive, and uninteresting, but requires accuracy; exactly the sort of thing that computers do well, and we don't.
- Matrices and graphs have rules and conventions: Sometimes these are just rules and conventions that help us communicate clearly. But sometimes the rules and conventions of the language of graphs and mathematics themselves lead us to see things in our data that might not have occurred to us to look for if we had described our data only with words.

So, we need to learn the basics of representing social network data using matrices and graphs. The next several chapters (3, 4, 5, and 6) introduce these basic tools.

[table of contents](#)

[table of context of the book](#)

Introduction to social network methods

3. Using graphs to represent social relations

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

Contents of chapter 3: Using graphs to represent social relations

- [Introduction: Representing networks with graphs](#)
 - [Graphs and sociograms](#)
 - [Kinds of graphs:](#)
 - [Levels of measurement](#): Binary, signed, and valued graphs
 - [Directed or "bonded" Ties](#) in the graph
 - [Simplex or multiplex relations](#) in the graph
 - [Summary](#)
 - [Study questions](#)
-

Introduction: Representing networks with graphs

Social network analysts use two kinds of tools from mathematics to represent information about patterns of ties among social actors: graphs and matrices. On this page, we will learn enough about graphs to understand how to represent social network data. On the next page, we will look at matrix representations of social relations. With these tools in hand, we can understand most of the things that network analysts do with such data (for example, calculate precise measures of "relative density of ties").

There is a lot more to these topics than we will cover here; mathematics has whole sub-fields devoted to "graph theory" and to "matrix algebra." Social scientists have borrowed just a few things that they find helpful for describing and analyzing patterns of social relations.

A word of warning: there is a lot of specialized terminology here that you do need to learn. its worth the effort, because we can represent some important ideas about social structure in quite simple ways, once the basics have been mastered.

[table of contents](#)

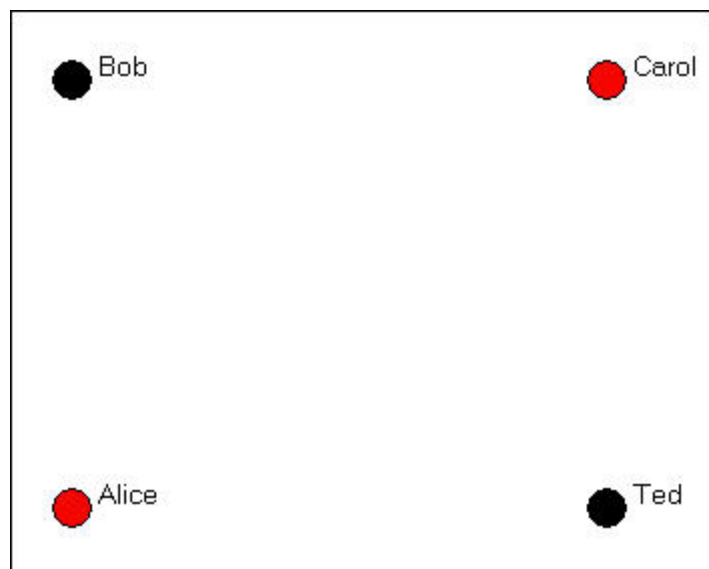
Graphs and Sociograms

There are lots of different kinds of "graphs." Bar-charts, pie-charts, line and trend charts, and many other things are called graphs and/or graphics. Network analysis uses (primarily) one kind of graphic display that consists of points (or nodes) to represent actors and lines (or edges) to represent ties or relations. When sociologists borrowed this way of graphing things from the mathematicians, they re-named their graphics "socio-grams." Mathematicians know the kind of graphic displays by the names of "directed graphs" "signed graphs" or simply "graphs."

There are a number of variations on the theme of socio-grams, but they all share the common feature of using a labeled circle for each actor in the population we are describing, and line segments between pairs of actors to represent the observation that a tie exists between the two. Let's suppose that we are interested in

summarizing who nominates whom as being a "friend" in a group of four people (Bob, Carol, Ted, and Alice). We would begin by representing each actor as a "node" with a label (sometimes nodes are represented by labels in circles or boxes). Figure 3.1 shows a graph with four labeled nodes, but no connections.

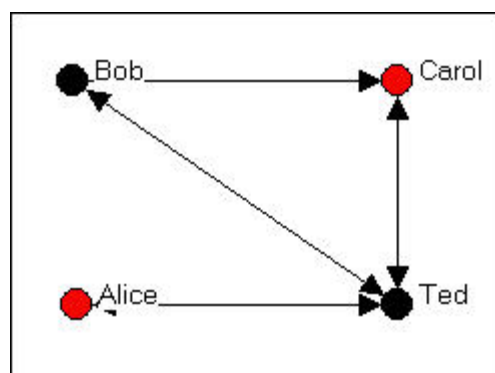
Figure 3.1. Nodes for a simple graph



In this example, we've also indicated an "attribute" of each actor by coloring the node (black for males, red for females). Coloring, shading, or different shapes and sizes are often used to represent attributes of the individual nodes.

We collected our data about friendship ties by asking each member of the group (privately and confidentially) who they regarded as "close friends" from a list containing each of the other members of the group. Each of the four people could choose none to all three of the others as "close friends." As it turned out, in our (fictitious) case, Bob chose Carol and Ted, but not Alice; Carol chose only Ted; Ted chose Bob and Carol and Alice; and Alice chose only Ted. We would represent this information by drawing an arrow from the chooser to each of the chosen, as in figure 3.2.

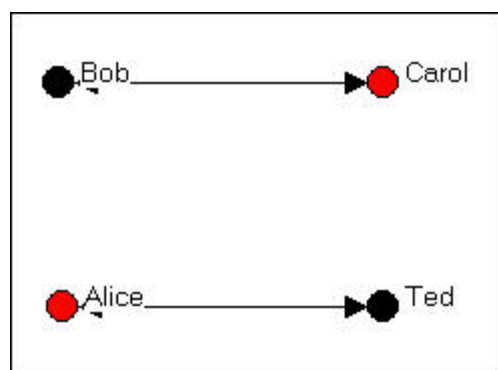
Figure 3.2. A directed graph of friendship ties



To reduce visual clutter, a double-headed arrow has been used when the relationship between two nodes is "reciprocated" (i.e. each actor chooses the other).

Let's suppose that we had also taken note of a second kind of relation - whether persons share the relationship "spouse" with one another. In our example, Bob and Carol are spouses, as are Ted and Alice. We can also represent this kind of a "bonded tie" with a directed graph as in figure 3.3.

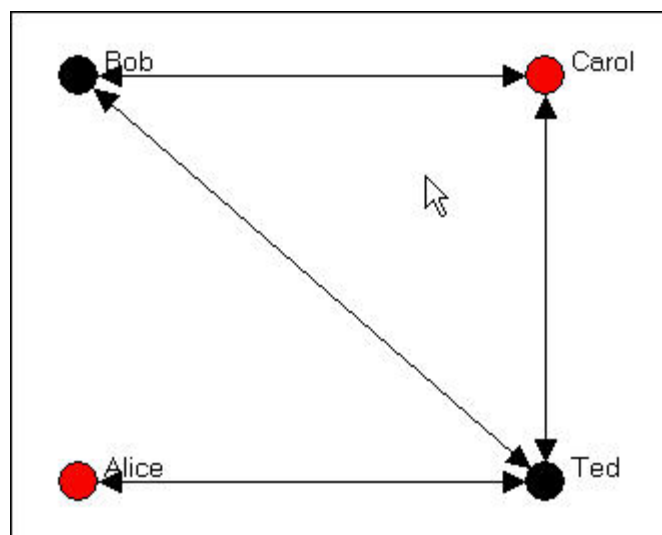
Figure 3.3. A directed graph of spousal ties



Where a tie is necessarily reciprocated (see the discussion of "bonded ties, below), a "simple" graph is often used instead of a "directed" graph. In a simple graph, relations are simply present or absent, and the relations are indicated by lines without arrow heads.

We can also represent multiple relations (multiplex relations) using graphs -- though with larger numbers of actors or many relations, the results may not be very easy to read. Let's combine the graphs of both "friendship" and "spousal" relations, as in figure 3.4.

Figure 3.4. A directed graph of multiplex relations (friendship and spouse)



In this figure, a tie is shown between two nodes whenever there is either a friendship tie, or a spousal tie, or both. This helps us to see that Bob, Carol, and Ted form a "clique" (i.e. each is connected to each of the others), and Alice is a "pendant" (tied to the group by only one connection).

This particular way for drawing the multiplex relation, however, loses some information about which ties connect which actors. As an alternative, one might want to superimpose the two single-relation graphs, and show multiple lines (or different color lines, or some dashed lines) to show the different kinds of connections.

[table of contents](#)

Kinds of Graphs

Now we need to introduce some terminology to describe different kinds of graphs. Figure 3.2 is an example of a *binary* (as opposed to a signed or ordinal or valued) and *directed* (as opposed to a co-occurrence or co-

presence or bonded-tie) graph. Figure 3.3 is an example of a "co-occurrence" or "co-presence" or "bonded-tie" graph that is *binary* and *undirected* (or simple). The social relations being described here are also *simplex* (in figures 3.2 and 3.3). Figure 3.4 is an example of one method of representing *multiplex* relational data with a single graph.

Let's take a moment to review some of this terminology in a little more detail.

[table of contents](#)

Levels of Measurement: Binary, Signed, and Valued Graphs

In describing the pattern of who describes whom as a close friend, we could have asked our question in several different ways. If we asked each respondent "is this person a close friend or not," we are asking for a binary choice: each person is or is not chosen by each interviewee. Many social relationships can be described this way: the only thing that matters is whether a tie exists or not. When our data are collected this way, we can graph them simply: an arrow represents a choice that was made, no arrow represents the absence of a choice. But, we could have asked the question a second way: "for each person on this list, indicate whether you like, dislike, or don't care." We might assign a + to indicate "liking," zero to indicate "don't care" and - to indicate dislike. This kind of data is called "signed" data. The graph with signed data uses a + on the arrow to indicate a positive choice, a - to indicate a negative choice, and no arrow to indicate neutral or indifferent. Yet another approach would have been to ask: "rank the three people on this list in order of who you like most, next most, and least." This would give us "rank order" or "ordinal" data describing the strength of each friendship choice. Lastly, we could have asked: "on a scale from minus one hundred to plus one hundred - where minus 100 means you hate this person, zero means you feel neutral, and plus 100 means you love this person - how do you feel about...". This would give us information about the value of the strength of each choice on a (supposedly, at least) ratio level of measurement. With either an ordinal or valued graph, we would put the measure of the strength of the relationship on the arrow in the diagram.

[table of contents](#)

Directed or "bonded" ties in the graph

In our example, we asked each member of the group to choose which others in the group they regarded as close friends. Each person (ego) then is being asked about ties or relations that they themselves direct toward others (alters). Each alter does not necessarily feel the same way about each tie as ego does: Bob may regard himself as a good friend to Alice, but Alice does not necessarily regard Bob as a good friend. It is very useful to describe many social structures as being composed of "directed" ties (which can be binary, signed, ordered, or valued). Indeed, most social processes involve sequences of directed actions. For example, suppose that person A directs a comment to B, then B directs a comment back to A, and so on. We may not know the order in which actions occurred (i.e. who started the conversation), or we may not care. In this example, we might just want to know that "A and B are having a conversation." In this case, the tie or relation "in conversation with" necessarily involves both actors A and B. Both A and B are "co-present" or "co-occurring" in the relation of "having a conversation." Or, we might also describe the situation as being one of an the social institution of a "conversation" that by definition involves two (or more) actors "bonded" in an interaction (Berkowitz).

"Directed" graphs use the convention of connecting nodes or actors with arrows that have arrow heads, indicating who is directing the tie toward whom. This is what we used in the graphs above, where individuals (egos) were directing choices toward others (alters). "Simple" or "Co-occurrence" or "co-presence" or "bonded-tie" graphs use the convention of connecting the pair of actors involved in the relation with a simple line segment (no arrow head). Be careful here, though. In a directed graph, Bob could choose Ted, and Ted

choose Bob. This would be represented by headed arrows going from Bob to Ted, and from Ted to Bob, or by a double-headed arrow. But, this represents a different meaning from a graph that shows Bob and Ted connected by a single line segment without arrow heads. Such a graph would say "there is a relationship called close friend which ties Bob and Ted together." The distinction can be subtle, but it is important in some analyses.

[table of contents](#)

Simplex or multiplex relations in the graph

Social actors are often connected by more than one kind of relationship. In our simple example, we showed two graphs of simple (sometimes referred to as "simplex" to differentiate from "multiplex") relations. The friendship graph (figure 3.2) showed a single relation (that happened to be binary and directed). The spouse graph (figure 3.3) showed a single relation (that happened to be binary and un-directed). Figure 3.4 combines information from two relations into a "multiplex" graph.

There are, potentially, different kinds of multiplex graphs. We graphed a tie if there was either a friendship or spousal relation. But, we could have graphed a tie only if there were both a friendship and spousal tie (what would such a graph look like?).

We also combined the information about multiple ties into a single line. Alternatively, one might use different symbols, colors, line widths, or other devices to keep all of the information about multiple relations visible in a multiplex graph -- but the result can often be too complicated to be useful.

[table of contents](#)

Summary

A graph (sometimes called a sociogram) is composed of nodes (or actors or points) connected by edges (or relations or ties). A graph may represent a single type of relations among the actors (simplex), or more than one kind of relation (multiplex). Each tie or relation may be directed (i.e. originates with a source actor and reaches a target actor), or it may be a tie that represents co-occurrence, co-presence, or a bonded-tie between the pair of actors. Directed ties are represented with arrows, bonded-tie relations are represented with line segments. Directed ties may be reciprocated (A chooses B and B chooses A); such ties can be represented with a double-headed arrow. The strength of ties among actors in a graph may be nominal or binary (represents presence or absence of a tie); signed (represents a negative tie, a positive tie, or no tie); ordinal (represents whether the tie is the strongest, next strongest, etc.); or valued (measured on an interval or ratio level). In speaking the position of one actor or node in a graph to other actors or nodes in a graph, we may refer to the focal actor as "ego" and the other actors as "alters."

[table of contents](#)

Review questions

1. What are "nodes" and "edges"? In a sociogram, what is used for nodes? for edges?
2. How do valued, binary, and signed graphs correspond to the "nominal" "ordinal" and "interval" levels of measurement?
3. Distinguish between directed relations or ties and "bonded" relations or ties.

4. How does a reciprocated directed relation differ from a "bonded" relation?
5. Give an example of a multi-plex relation. How can multi-plex relations be represented in graphs?

Application questions

1. Think of the readings from the first part of the course. Did any studies present graphs? If they did, what kinds of graphs were they (that is, what is the technical description of the kind of graph or matrix). Pick one article and show what a graph of its data would look like.
2. Suppose that I was interested in drawing a graph of which large corporations were networked with one another by having the same persons on their boards of directors. Would it make more sense to use "directed" ties, or "bonded" ties for my graph? Can you think of a kind of relation among large corporations that would be better represented with directed ties?
3. Think of some small group of which you are a member (maybe a club, or a set of friends, or people living in the same apartment complex, etc.). What kinds of relations among them might tell us something about the social structures in this population? Try drawing a graph to represent one of the kinds of relations you chose. Can you extend this graph to also describe a second kind of relation? (e.g. one might start with "who likes whom?" and add "who spends a lot of time with whom?").
4. Make graphs of a "star" network, a "line" and a "circle." Think of real world examples of these kinds of structures where the ties are directed and where they are bonded, or undirected. What does a strict hierarchy look like? What does a population that is segregated into two groups look like?

[table of contents](#)

[table of contents of the book](#)

Introduction to social network methods

4. Working with NetDraw to visualize graphs

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

Contents of this chapter:

- [Introduction: A picture is worth...](#)
- [Node attributes](#)
- [Relation properties](#)
- [Location, location, location](#)
- [Highlighting parts of the network](#)
- [A few hints on data handling with NetDraw](#)

Introduction: A picture is worth...

As we saw in chapter 3, a graph representing the information about the relations among nodes can be an very efficient way of describing a social structure. A good drawing of a graph can immediately suggest some of the most important features of overall network structure. Are all the nodes connected? Are there many or few ties among the actors? Are there sub-groups or local "clusters" of actors that are tied to one another, but not to other groups? Are there some actors with many ties, and some with few?

A good drawing can also help us to better understand how a particular "ego" (node) is "embedded" (connected to) its "neighborhood" (the actors that are connected to ego, and their connections to one another) and to the larger graph (is "ego" an "isolate" a "pendant"?). By looking at "ego" and the "ego network" (i.e. "neighborhood"), we can get a sense of the structural constraints and opportunities that an actor faces; we may be better able to understand the role that an actor plays in a social structure.

There is no single "right way" to represent network data with graphs. There are a few basic rules, and we reviewed these in the previous chapter. Different ways of drawing pictures of network data can emphasize (or obscure) different features of the social structure. It's usually a good idea to play with visualizing a network, to experiment and be creative. There are a number of software tools that are available for drawing graphs, and each has certain strengths and limitations. In this chapter, we will look at some commonly used techniques for visualizing graphs using NetDraw (version 4.14, which is distributed along with UCINET). There are many other packages though, and you might want to explore some of the tools available in Pajek, and Mage (look for software at the web-site of the International Network of Social Network Analysts - INSNA).

Of course, if there are a large number of actors or a large number of relations among them, pictures may not help the situation much; numerical indexes describing the graph may be the only choice. Numerical approaches and graphical approaches can be used in combination, though. For example, we might first calculate the "between-ness centrality" of the nodes in a large network, and then use graphs that include only those actors that have been identified as "important."

[table of contents](#)

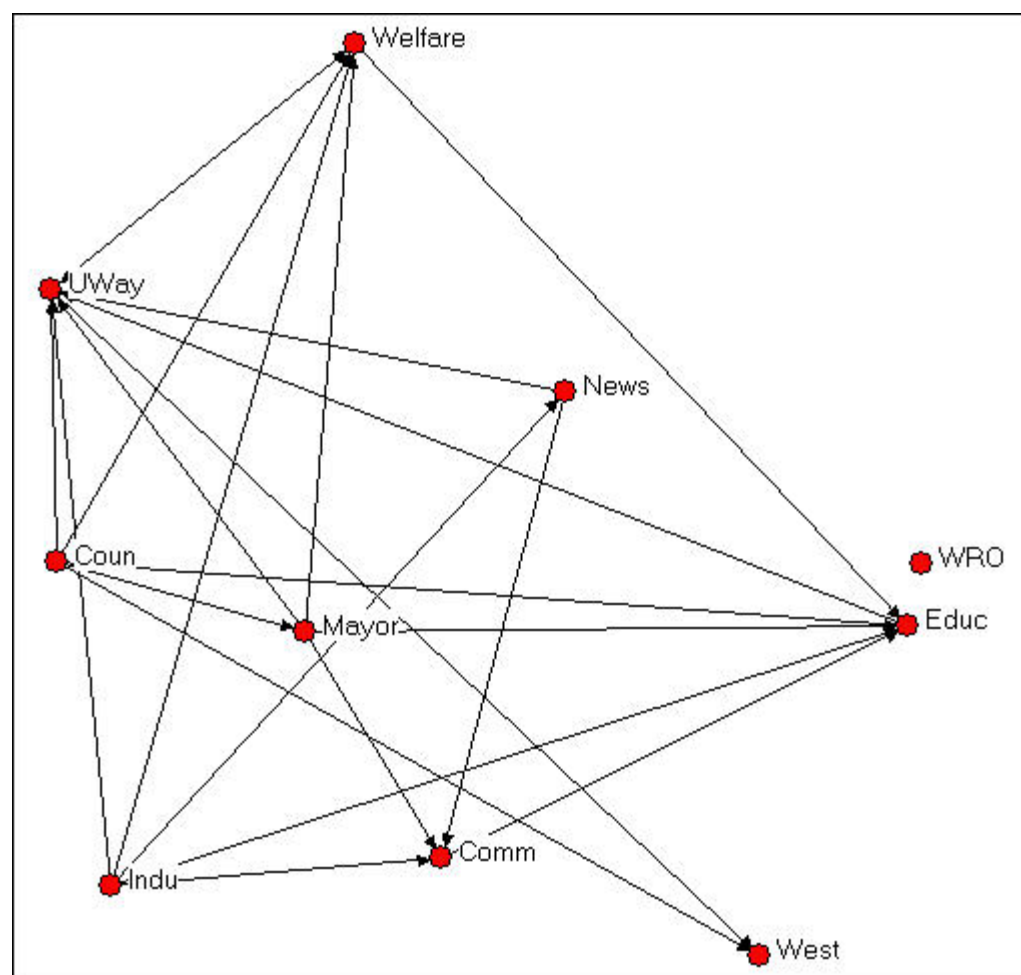
Node attributes

Differences of kind: We often have information available about some attributes of each the actors in our network. In the Bob, Carol, Ted and Alice example, we noted that two of the actors were male and two female. The scores of the cases (Bob, Carol, Ted, Alice) on the variable "sex" are a nominal dichotomy. It is also pretty common to be able to divide actors in a "multiple-choice" way; that is, we can record an attribute as a nominal polyotomy (for example, if we knew the religious affiliation of each actor, we might record it as "Christian," "Muslim," "Jewish," "Zoroastrian," or whatever).

It is often the case that the structure of a network depends on the attributes of the actors embedded in it. If we are looking at the network of "spouse" ties among Bob, Carol, Ted, and Alice, one would note that ties exist for male-female pairs, but not (in our example) for female-female or male-male pairs. Being able to visualize the locations of different types of actors in a graph can help us see patterns, and to understand the nature of the social processes that generated the tie structure.

Using colors and shapes are useful ways of conveying information about what "type" of actor each node is. Figures 4.1 and 4.2 provide an example. The data here describe the exchange of information among ten organizations that were involved in the local political economy of social welfare services in a Midwestern city (from a study by David Knoke; the data are one of the data sets distributed with UCINET). In Figure 4.1, NetDraw has been used to render a directed graph of the data. This is done by opening [Netdraw>File>Open>UCInet dataset>Network](#), and locating the data file. NetDraw produces a basic graph that you can then edit.

Figure 4.1. Knoke information exchange network



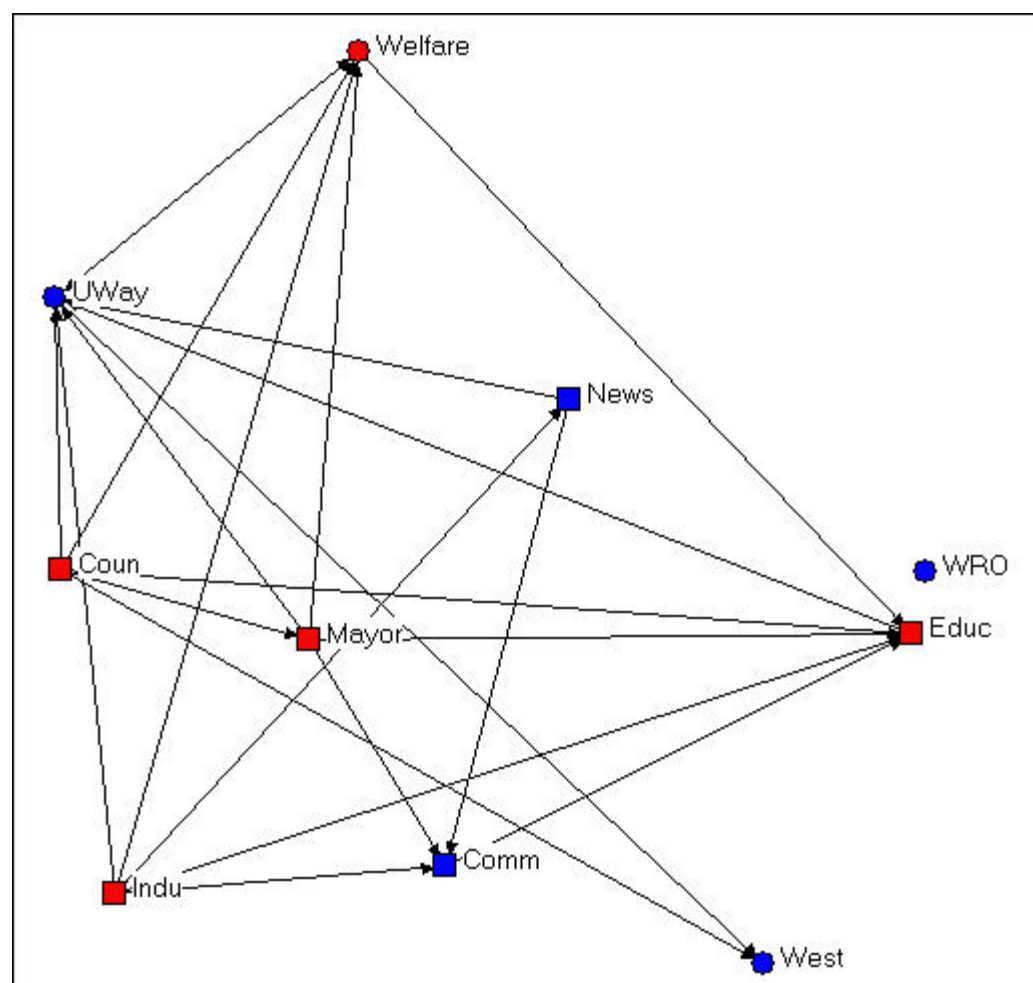
Institutional theory might suggest that information exchange among organizations of the same "type" would be more common than information exchange between organizations of different types. Some of the organizations here are governmental (Welfare, Coun, Educ, Mayor, Indu), some are non-governmental (UWay, News, WRO, Comm, West).

In Netdraw, we used the [Transform>mode attribute](#) editor to assign a score of "1" to each node if it was governmental, and "0" if it was not. We then used [Properties>nodes>color>attribute-based](#) to select the government attribute, and assign the color red to government organizations, and blue to non-government organizations. You could also create an attribute data file in UCINET using the same nodes as the network data file, and creating one or more columns of attributes. [NetDraw>File>Open>UCInet dataset>Attribute data](#) can then be used to open the attributes, along with the network, in NetDraw.

Ecological theory of organizations suggests that a division between organizations that are "generalists" (i.e. perform a variety of functions and operate in several different fields) and organizations that are "specialists" (e.g. work only in social welfare) might affect information-sharing patterns.

In Netdraw, we used the [Transform>mode attribute](#) editor to create a new column to hold information about whether each organization was a "generalist" or a "specialist." We assigned the score of "1" to "generalists" (e.g. the Newspaper, Mayor) and a score of "0" to "specialists" (e.g. the Welfare Rights Organization). We then used [Properties>nodes>shape>attribute-based](#) to assign the shape "square" to generalists and "circle" to specialists. The result of these operations is shown in figure 4.2.

Figure 4.2. Knoke information exchange with government/non-government and specialist/generalist codes

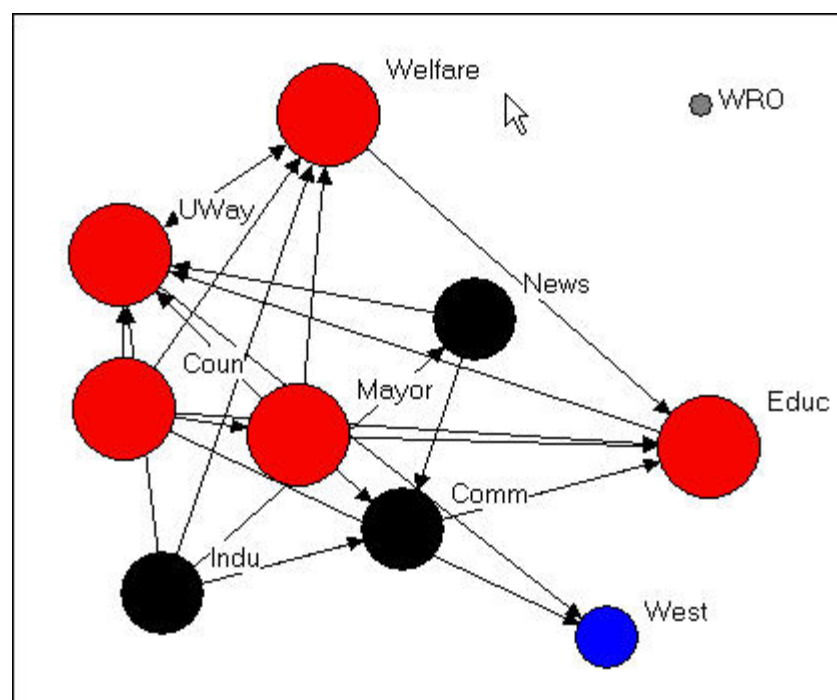


A visual inspection of the diagram with the two attributes highlighted by node color and shape is much more

informative about the hypotheses of differential rates of connection among red/blue and among circle/square. It doesn't look like this diagram is very supportive of either of our hypotheses.

Identifying types of nodes according to their attributes can be useful to point out characteristics of actors that are based on their position in the graph. Consider the example in the next figure, 4.3.

Figure 4.3 Knoke information exchange with k-cores



This figure was created by using the [Analysis>K-core](#) tool that is built into NetDraw. We'll talk about the precise definition of a k-core in a later chapter. But, generally, a k-core is a set of nodes that are more closely connected to one another than they are to nodes in other k-cores. That is, a k-core is one definition of a "group" or "sub-structure" in a graph.

Figure 4.3 shows four sub-groups, which are colored to identify which nodes are members of which group (the "West" group and the "WRO" group each contain only a single node). In addition, the size of the nodes in each K-core are proportional to the size of the K-core. The largest group contains government members (Mayor, County Government, Board of Education), as well as the main public (Welfare) and private (United Way) welfare agencies. A second group, colored in black, groups together the newspaper, chamber of commerce, and industrial development agency. Substantively, this actually makes some sense!

This example shows that color and shape of nodes to represent qualitative differences among actors can be based on classifying actors according to their position in the graph, how they are embedded, rather than on some inherent feature of the actor itself (e.g. governmental or non-governmental). One can use UCINET (or other programs) to identify "types" of actors based on their relations (e.g. where are the cliques?), and then enter this information into the attribute editor of NetDraw ([Transform>node attribute editor>edit>add column](#)). The groupings that are created by using [Analysis](#) tools already built-in to NetDraw are automatically added to the node attribute data base).

Differences of amount: Figure 4.3 also uses the size of the nodes ([Properties>nodes>size>attribute-based](#)) to display an index of the number of nodes in each group. This difference of amount among the nodes is best indicated, visually, by assigning the size of the node to values of some attribute. In example 4.3, NetDraw has done this automatically for an amount that was computed by its [Analysis>K-cores](#) tool. One can easily compute other variables reflecting differences in amount among actors (e.g. how many "out

degrees" or arrows from each actor are there?) using UCINET or other programs. Once these quantities are computed, they can be added to NetDraw ([Transform>node attribute editor>edit>add column](#)), and then added to the graph ([Properties>nodes>size>attribute-based](#)).

Differences of amount among the nodes could also reflect an attribute that is inherent to an actor alone. In the welfare organizations example, we might know the annual budget or number of employees of each organization. These measures of organizational "size" might be added to the node attributes, and used to scale the size of the nodes so that we could see whether information sharing patterns differ by organizational size.

Color, shape, and size of the nodes in a graph then can be used to visualize information that we have about the attributes of the actors. These attributes can be based on either "external" information about inherent differences in kind and amount among the actors; or, the attributes can be based on "internal" information about differences in kind and amount that describe how the actor is embedded in the relational network.

[table of contents](#)

Relation properties

A graph, as we discussed in the last chapter, is made up of both the actors and the relations among the actors. The relations among the actors (the line segments in a simple graph or the arrows in a directed graph) can also have "attributes." Sometimes it can be very helpful to use color and size to indicate difference of kind and amount among the relations. You can be creative with this idea to explore and display patterns in the connections among the actors in a network. Here are a few ideas of things that you could do.

Suppose that you wanted to highlight certain "types" of relations in the graph. For example, in a sociogram of friendship ties you might want to show how the patterns of ties among persons of the same sex differed from the patterns of ties among persons of different sexes. You might want to show the ties in the graph as differing in color or shape (e.g. dashed or solid line) depending on the type of relation. If you have recorded the two kinds of relations (same sex, different sex) as two relations in a multiplex graph, then you can use NetDraw's [Properties>Lines>Color](#) from the menu. Then select [Relations](#), and choose the color for each of the relations you want to graph (e.g. red for same-sex, blue for different-sex).

Line colors (but not line shapes) can be used to highlight links within actors of the same type or between actors of different types, or both, using NetDraw. First select [Properties>Lines>Node-attribute](#) from the menus. Then select whether you want to color the ties among actors of the same attribute type ("[within](#)") or ties among actors of different types ("[between](#)"), or both. Then use the drop-down menu to select the attribute that you want to graph.

If you have measured each tie with an ordinal or interval level variable (usually reflecting the "strength" of the tie), you can also assign colors to ties based on tie strength ([Properties>Lines>Color>Tie-strength](#)). But, when you have information on the "value" of the relations, a different method would usually be preferred.

Where the ties among actors have been measured as a value (rather than just present-absent), the magnitude of the tie can be suggested by using thicker lines to represent stronger ties, and thinner lines to represent weaker ties. Recall that sometimes ties are measured as negative-neutral-positive (recorded as -1, 0, +1), as grouped ordinal (5=very strong, 4=strong, 3= moderate, 2=weak, 1=very weak, 0=absent), full-rank order (10=strongest tie of 10, 9=second strongest tie of 10, etc.), or interval (e.g. dollars of trade flowing from Belgium to the United States in 1975).

Since the value of the tie is already recorded in the data set, it is very easy to get NetDraw to visualize it in the graph. From the menus, select [Properties>Lines>Size](#). Then, select [Tie-Strength](#) and indicate which

relation you want graphed. You can select the amount of gradation in the line widths (e.g. from 0 to 5 for a grouped ordinal variable with 6 levels; or from 5 to 10 if you want really thick lines).

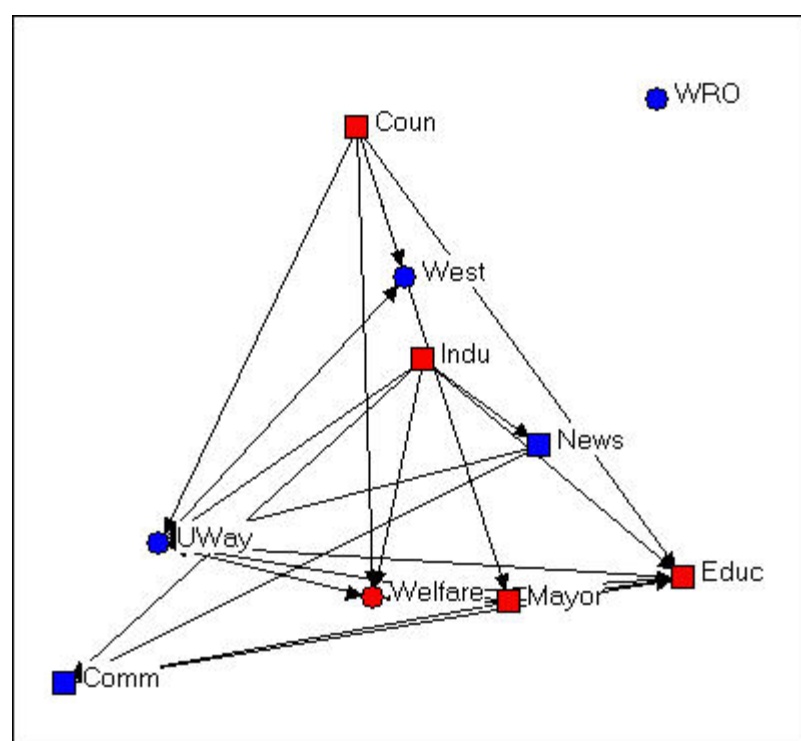
Using line colors and thickness, you can highlight certain types of ties and varying strength of ties among actors in the network. This can be combined with visual highlighting of attributes of the actors to make a compelling presentation of the features of the graph that you want to emphasize. There is a lot of art to this, and you will have to play and experiment. Node and line attributes can obscure as well as reveal; they can mis-represent, as well as represent. Sometimes, they add to the confusion of an already too-complicated graph.

[table of contents of this page](#)

Location, location, location

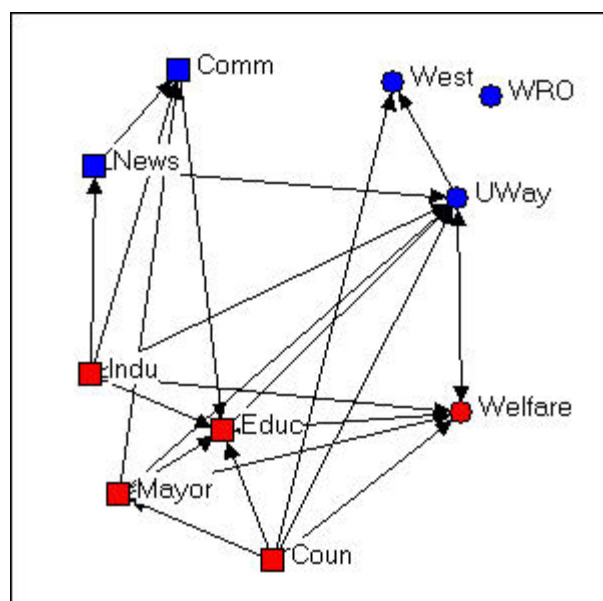
Most graphs of networks are drawn in a two-dimensional "X-Y axis" space (Mage and some other packages allow 3-dimensional rendering and rotation). Where a node or a relation is drawn in the space is essentially arbitrary -- the full information about the network is contained in its list of nodes and relations. The figures below are exactly the same network (Knoke's money flow network) that has been rendered in several different ways.

Figure 4.4 Random configuration of Knoke's money network



This drawing was created using NetDraw's [Layout>Random](#) on the graph that we had previously "colored" (blue for non-government, red for government; circles for welfare specialists, squares for generalists). This algorithm locates the nodes randomly in the X-Y space (you can use other tools to change the size of the graphic, rotate it, etc.). Since the X and Y directions don't "mean" anything, the location of the nodes and relations don't provide any particular insight.

Figure 4.5 Free-hand grouping by attributes configuration of Knoke's money network



In figure 4.5, we've used the "drag and drop" method ("grab" a node with the cursor, and drag it to a new location) to relocate the nodes so that organizations that share the same combinations of attributes are located in different quadrants of the graph. Since we had hypothesized that organizations of like "kind" would have higher rates of connection, this is a useful (but still arbitrary) way to locate the points. If the hypothesis were strongly supported (and its not) most of the arrows would be located within each the four quadrants, and there would be few arrows between quadrants. NetDraw has a built-in tool that allows the user to assign the X and Y dimensions of the graph to scores on attributes (either categorical or continuous): [Layout>Attributes as Coordinates](#), and then select attributes to be assigned to X or Y or both. This can be a very useful tool for exploring how patterns of ties differ within and between "partitions" (or types of nodes).

Figure 4.6 Circle configuration of Knoke's money network

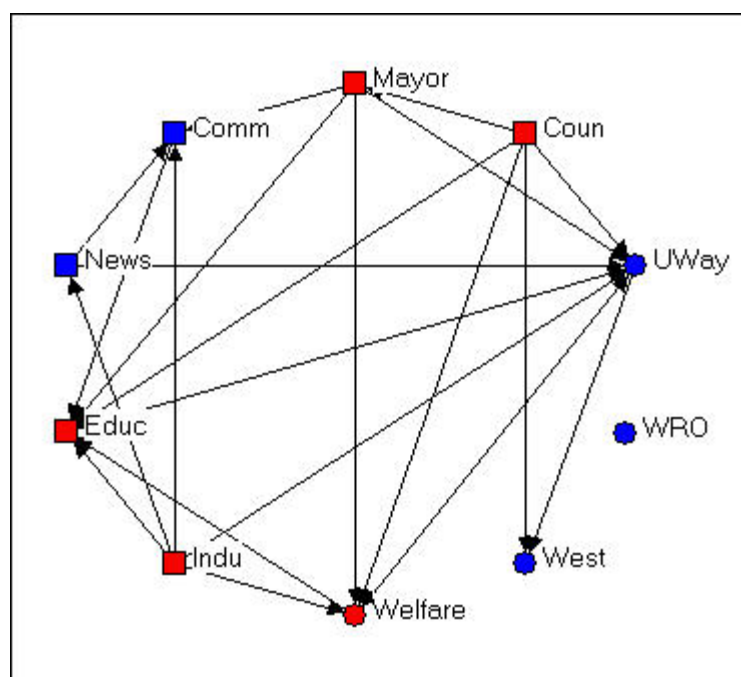


Figure 4.6 shows the same graph using [Layout>Circle](#), and selecting the "generalist-specialist" (i.e. the circle or square node type) as the organizing criterion. Circle graphs are commonly used to visualize which nodes are most highly connected. The nodes are located at equal distances around a circle, and nodes that are highly connected are very easy to quickly locate (e.g. UWay, Educ) because of the density of lines.

When nodes sharing the same attribute are located together in a segment of the circle, the density of ties within and between types is also quite apparent.

In each of the different layouts we've discussed so far, the distances between the nodes are arbitrary, and can't be interpreted in any meaningful way as "closeness" of the actors. And, the "directions" X and Y have no meaning -- we could rotate any of the graphs any amount, and it would not change a thing about our interpretation. There are several other commonly used graphic layouts that do try to make the distances and/or directions of locations among the actors somewhat more meaningful.

Figure 4.7 Non-metric multi-dimensional scaling configuration of Knoke's money network

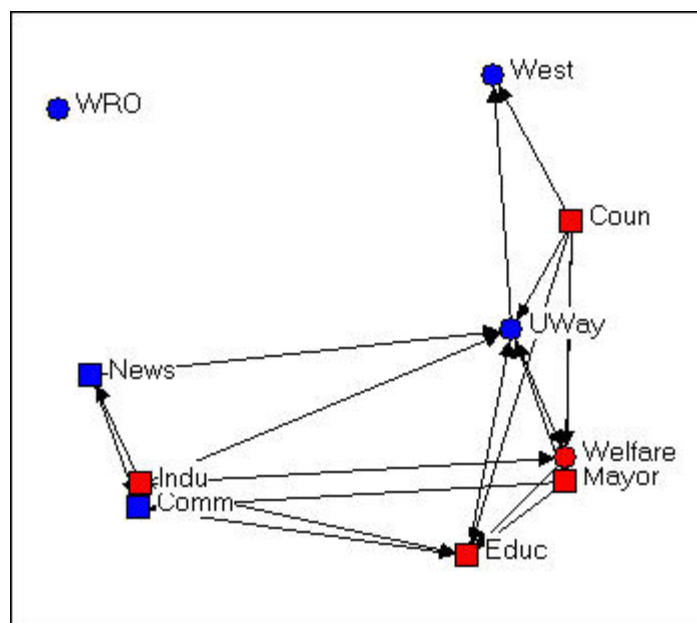


Figure 4.7 was generated using the [Layout>Graph Theoretic Layout>MDS](#) tool of NetDraw. MDS stands for (non-metric, in this case) "Multi-Dimensional Scaling." MDS is a family of techniques that is used (in network analysis) to assign locations to nodes in multi-dimensional space (in the case of the drawing, a 2-dimensional space) such that nodes that are "more similar" are closer together. There are many reasonable definitions of what it means for two nodes to be "similar." In this example, two nodes are "similar" to the extent that they have similar shortest paths (geodesic distances) to all other nodes. There are many, many ways of doing MDS, but the default tools chosen in NetDraw can often generate meaningful renderings of graphs that provide insights. NetDraw has several built-in algorithms for generating coordinates based on similarity (metric and non-metric two-dimensional scaling, and principle components analysis).

One very important difference between figure 4.7 and the earlier graphs is that the distances between the nodes is interpretable. The "Welfare" and "Mayor" nodes are very similar in their geodesic distances from other actors. "West" and "Educ" have very different patterns of ties to the other nodes.

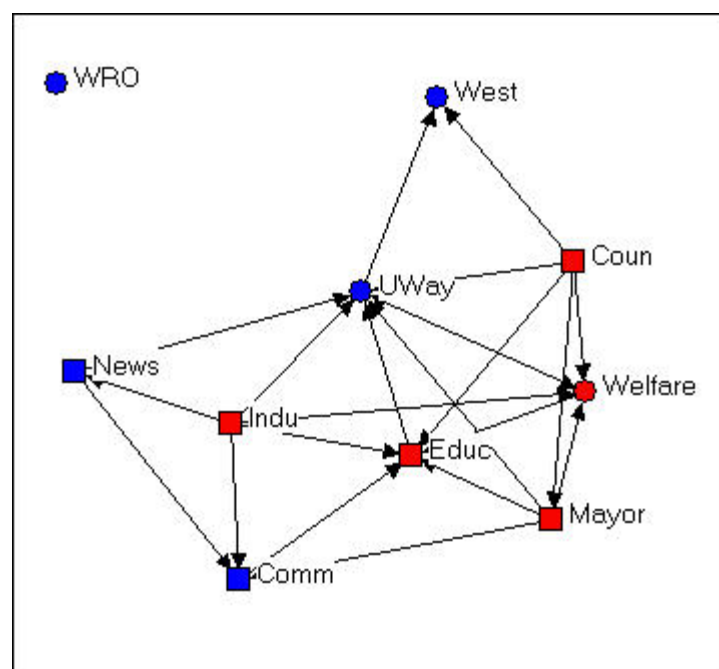
The other important difference between figure 4.7 and the earlier graphs is that direction may be meaningful. Notice that there is a cluster of nodes at the left (News, Indu, Comm) that are all pretty much not welfare organizations themselves, while the nodes at the right are (generally) more directly involved in welfare service provision. The upper left-hand quadrant contains mostly "blue" nodes, while the lower right quadrant contains mostly "red" ones -- so one "direction" might be interpreted as "non-government/government."

Let me emphasize that different applications of MDS (and other scaling tools) to different definitions of what it means for nodes to be "similar" can generate wildly different looking graphs. These are (almost always) exploratory techniques, and there is (usually) no single "correct" interpretation. But, a graphic that uses both distance and direction to summarize something about the structure of the network can provide considerable

insight when compared to graphs (like figures 4.4, 4.5, and 4.6) that don't.

Consider one last example of rendering the same data, this time using NetDraw's unique built-in algorithm for locating points ([Layout>Graph Theoretic Layout>Spring Embedding](#)).

Figure 4.8 "Spring-embedding" configuration of Knoke's money network



You might immediately notice that this graph is fairly similar to the MDS solution. The algorithm uses iterative fitting (i.e. start with a random graph, measure "badness" of fit; move something, measure "badness" and if it's better keep going in that direction...) to locate the points in such a way as to put those with smallest path lengths to one another closest in the graph. This approach can often locate points very close together, and make for a graph that is hard to read. In the current example, we've also selected the optional "node repulsion" criterion that creates separation between objects that would otherwise be located very close to one another. We've also used the optional criterion of seeking to make the paths of "equal edge length" so that the distances between adjacent objects are similar.

The result is a graph that preserves many of the features of the dimensional scaling approach (distances are still somewhat interpretable; directions are often interpretable), but is usually easier to read -- particularly if it matters which specific nodes are where (rather than node types or clusters).

There is no one "right way" to use space in a graph. But one can usually do much better than a random configuration -- particularly if one has some prior hypotheses or research questions about the kinds of patterns that would be meaningful.

[table of contents of this page](#)

Highlighting parts of the network

Large networks (those that contain many actors, many kinds of relations, and/or high densities of ties) can be very difficult to visualize in any useful way -- there is simply too much information. Often, we need to clear away some of the "clutter" to see main patterns more clearly.

One of the most interesting features of social networks -- whether small or large -- is the extent to which we

can locate "local sub-structures." We will discuss this topic a good bit more in a later chapter. Highlighting or focusing attention on sub-sets of nodes in a drawing can be a powerful tool for visualizing sub-structures.

In this section, we will briefly outline some approaches to rendering network drawings that can help to simplify complex diagrams and locate interesting sub-graphs (i.e. collections of nodes and their connections).

Clearing away the underbrush

Social structures can be composed of multiple relations. Bob, Carol, Ted, and Alice in our earlier example are a multi-plex structure of people connected by both friendship and by spousal ties. Graphs that display all of the connections among a set of nodes can be very useful for understanding how actors are tied together -- but they can also get so complicated and dense that it is difficult to see any patterns. There are a couple approaches that can help.

One approach is to combine multiple relations into an index. For example, one could combine the information on friendship and spousal ties using an "and" rule: if two nodes have both a friendship and spousal tie, then they have a tie - otherwise they do not (i.e. if they have no tie, or only one type of tie). Alternatively, we could create an index that records a tie when there is either a friendship tie or a spousal tie. If we had measured relations with values, rather than simple presence-absence, multiple relations could be combined by addition, subtraction, multiplication, division, averaging, or other methods. UCINET has tools for these kinds of operations, that are located at: [Transform>matrix operations>within dataset>aggregations](#).

The other approach is to simplify the data a bit. NetDraw has some tools that can be of some help.

Rather than examining the information on multiple kinds of ties in one diagram, one can look at them one at a time, or in combination. If the data have been stored as a UCINET or NetDraw data file with multiple relations, then the [Options>View>Relations Box](#) opens a dialog box that lets you select which relations you want to display. Suppose that we had a data set in which we had recorded the friendship ties among a number of people at intervals over a period of time. By first displaying the first time point, and then adding subsequent time point, we can visualize the evolution of the friendship structure.

It isn't unusual for some of the nodes in a graph of a social network to not be connected to the others at all. Nodes that aren't connected are called "isolates." Some nodes may be connected to the network by a single tie. These nodes sort of "dangle" from the diagram; they are called "pendants." One way of simplifying graphs is to hide isolates and/or pendants to reduce visual clutter. Of course, this does mis-represent the structure, but it may allow us to focus more attention where most of the action is. NetDraw has both button-bar tools and a menu item ([Analysis>Isolates](#)) to hide these less-connected nodes.

Finding and visualizing local sub-structures

One of the common questions in network analysis is whether a graph displays various kinds of "sub-structures." For example, a "clique" is a sub-structure that is defined as a set of nodes where every element of the set is connected to every other member. A network that has no cliques might be a very different place than a network that has many small cliques, or one that has one clique and many nodes that are not part of the clique. We'll take a closer look at UCINET tools for identifying sub-structures in a later chapter.

NetDraw has built-in a number of tools for identifying sub-structures, and automatically coloring the graph to identify them visually.

[Analysis>components](#) locates the parts of graph that are completely disconnected from one another, and colors each set of nodes (i.e. each component). In our Bob-Carol-Ted-Alice example, the entire graph is one component, because all the actors are connected. In the welfare bureaucracies example, there are two

components, one composed of only WRO (which does not receive ties from any other organization) and the other composed of the other nine nodes. In NetDraw, executing this command also creates a variable in the database of node attributes -- as do all the other commands discussed here. These attributes can then be used for selecting cases, changing color, shape, and size, etc.

Analysis>Blocks and Cutpoints locates parts of the graph that would become disconnected components if either one node or one relation were removed (the blocks are the resulting components; the cutpoint is the node that would, if removed, create the dis-connect). NetDraw graphs these sub-structures, and saves the information in the node-attribute database.

Analysis>K-cores locates parts of the graph that form sub-groups such that each member of a sub-group is connected to N-K of the other members. That is, groups are the largest structures in which all members are connected to all but some number (K) of other members. A "clique" is a group like this where all members are connected to all other members; "fuzzier" or "looser" groups are created by increasing "K." NetDraw identifies the K-cores that are created by different levels of K, and provides colored graphs and data-base entries.

Analysis>Subgroups>block based. Sorry, but I don't know what this algorithm does! Most likely, it creates sub-structures that would become components with differing amounts of nodes/relations removed.

Analysis>Subgroups>Hierarchical Clustering of Geodesic Distances. The geodesic distance between two nodes is the length of the shortest path between them. A hierarchical clustering of distances produces a tree-like diagram in which the two nodes that are most similar in their profile of distances to all other points are joined into a cluster; the process is then repeated over and over until all nodes are joined. The resulting graphic is one way of understanding which nodes are most similar to one another, and how the nodes may be classified into "types" based on their patterns of connection to other nodes. The graph is colored to represent the clusters, and database information is stored about the cluster memberships at various levels of aggregation. A hierarchical clustering can be very interesting in understanding which groups are more homogeneous (those that group together at early stages in the clustering) than others; moving up the clustering tree diagram, we can see a sort of a "contour map" of the similarity of nodes.

Analysis>Subgroups>Factions (select number). A "faction" is a part of a graph in which the nodes are more tightly connected to one another than they are to members of other "factions." This is quite an intuitively appealing idea of local clustering or sub-structure (though, as you can see, only one such idea). NetDraw asks you how many factions you would like to find (always explore various reasonable possibilities!). The algorithm then forms the number of groups that you desire by seeking to maximize connection within, and minimize connection between the groups. Points are colored, and the information about which nodes fall in which partitions (i.e. which cases are in which factions) is saved to the node attributes database.

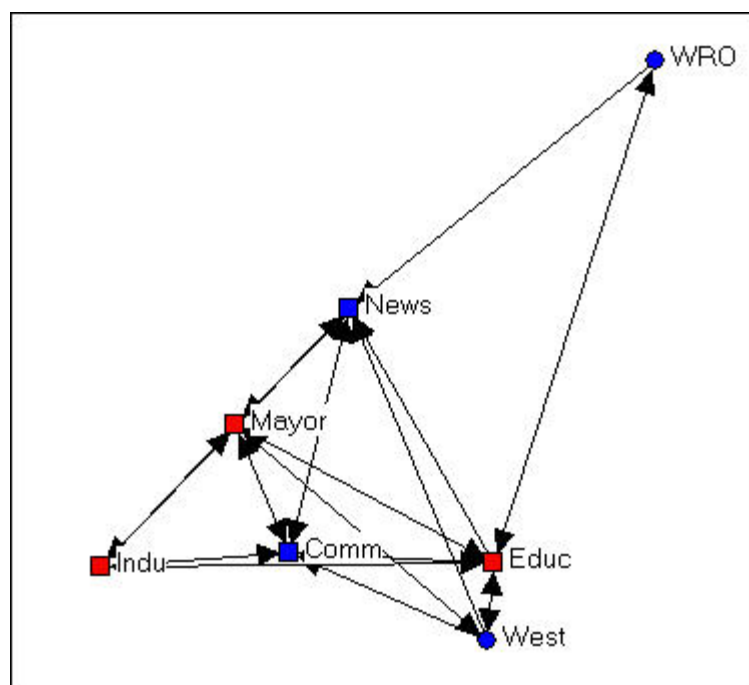
Analysis>Subgroups>Newman-Girvan. This is another numerical algorithm that seeks to create clusters of nodes that are closely connected within, and less connected between clusters. The approach is that of "block modeling." Rows and columns are moved to try to create "blocks" where all connections within a block are present, and all connections between blocks are absent. This algorithm will usually produce results similar to the factions algorithm. Importantly, though, the Newman-Girvan algorithm also produces measures of goodness-of-fit of the configuration for two blocks, three blocks, etc. This allows you to get some sense of what division into blocks is optimal for your needs (there isn't one "right" answer).

Ego Networks (neighborhoods)

A very useful way of understanding complicated network graphs is to see how they arise from the local connections of individual actors. The network formed by selecting a node, including all actors that are connected to that node, and all the connections among those other actors is called the "ego network" or (1-

step) neighborhood of an actor. Figure 4.9 is an example from the Knoke bureaucracies information network, where we select as our "ego" the board of education.

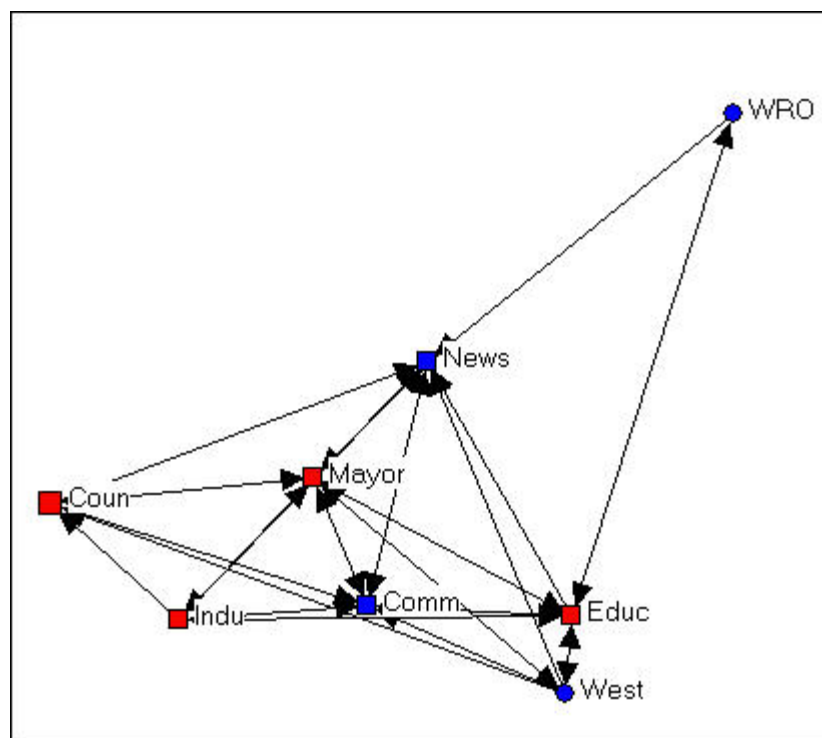
Figure 4.9. Ego network of Educ in Knoke information network



We note that the ego-network of the board of education is fairly extensive, and that the density of connection among the actors in the ego-network is fairly high. This is to say the the board of education is quite "embedded" in a dense local sub-structure.

Next, let's add the ego network of the "West" agency, in figure 4.10.

Figure 4.10. Ego networks of Educ and West in Knoke information network



The two ego networks combined result in a fully connected structure. We note that one connection between Educ and Coun is mediated by West.

One can often gain considerable insight about complicated networks by "building" them starting with one actor and adding others. Or, one can begin with the whole network, and see what happens as individual's ego networks are removed.

The network of each individual actor may also be of considerable interest. Who's most connected? How dense are the neighborhoods of particular actors?

NetDraw has useful tools for visualizing and working with ego-networks. The [Layout>Egonet](#) command presents a dialog box that lets you select which ego's networks are to be displayed. You can start with all the actors and delete; or start with focal actors and build up the full network.

[table of contents of this page](#)

A few hints on data handling with NetDraw

Input

There are several ways to get data into NetDraw. Probably the simplest is to import data from UCINET or Pajek. The [File>Open](#) command lets you read a UCINET text (DL) file (discussed elsewhere), and existing UCINET dataset, or a Pajek dataset. This menu also is used to access data that have been stored in the native data format of the NetDraw program (.VNA format). Once the data has been imported with the [Open](#) command, the node and line attribute editors of NetDraw can be used to create a diagram that can be saved with colors, shapes, locations, etc.

A second method is to build a dataset within NetDraw itself. Begin by creating a random network ([File>Random](#)). This creates an arbitrary network of 20 nodes. You can then use the node attributes editor ([Transform>Node Attribute editor](#)) and the link editor ([Transform>Link Editor](#)) to modify the nodes (and add or delete nodes) and their attributes; and to create connections among nodes. This is great for small, simple networks; for more complicated data, it's best to create the basic data set elsewhere and import it.

The third method is to use an external editor to create a NetDraw dataset (a .vna file) directly. This file is a plain ascii text file (if you use a word processor, be sure to save as ascii text). The contents of the file is pretty simple, and is discussed in the brief tutorial to NetDraw. Here is part of the file for the Knoke data, after we have created some of the diagrams we've seen.

```
*Node data
"ID", "General", "Size", "Govt"
  "1" "1" "3" "1"
  "2" "1" "2" "0"
.
.
  "9" "0" "2" "1"
  "10" "0" "1" "0"
*Node properties
ID x y color shape size shortlabel labelsize labelcolor active
"1" 51 476 255 2 16 "Coun" 11 0 TRUE
"2" 451 648 16711680 2 11 "Comm" 11 0 TRUE
.
.
"9" 348 54 255 1 11 "Welfare" 11 0 TRUE
"10" 744 801 16711680 1 6 "West" 11 0 TRUE
*Tie data
FROM TO "KNOKI" "KNOKM"
```

```

"1" "2" 1 0
"1" "5" 1 1
.
.
"9" "3" 0 1
"9" "8" 0 1
*Tie properties
FROM TO color size active
"1" "2" 0 1 TRUE
"1" "5" 0 1 TRUE
.
.
"9" "3" 0 1 FALSE
"9" "8" 0 1 FALSE

```

There are four sections of code here (not all are needed, but the `*node data` and `*tie data` are, to define the network structure). `*Node data` lists variables describing the nodes. An ID variable is necessary, the other variables in the example describe attributes of each node. The (optional) `*Node properties` section lists the variables, and gives values for ID, location on the diagram (X and Y coordinates from the upper left corner), shape, size, color, etc. Usually, one will not create this code; rather you input the data, use NetDraw to create a diagram, and save the result as a file -- and this section (and the `*Tie properties`) is created for you.

The `*Tie data` section is necessary to define the connections among the nodes. There is one data line for each relation in the graph. Each data line is described by its origin and destination, and value. Here, since there are two relations, "KNOKI" and "KNOKM" there are two values -- each of which happens, in our example, to be binary (but they could be valued).

The `*Tie properties` section is probably best created by using NetDraw and saving the resulting file. Each tie is identified by origin and destination, and its color and size are set. Here, certain ties are not to be visible in the drawing (the "active" property is set to "FALSE").

Output

When you are working with NetDraw, it is a good idea to save a copy of your work in the format (.vna, above) that is native to the program ([File>Save Data As>Vna](#)). This format keeps all of the information about your diagram (what's visible and not, node and line attributes, locations) so that you can re-open the diagram looking exactly as you left it.

You may also want to save datasets created with NetDraw to other program's formats. You won't be able to save all of the information about node and line properties and locations, but you can save the basic network (what are the nodes, which is connected to which) and node attributes. [File>Save Data As>Pajek](#) lets you save the network, partitions of it (which record attributes), and clusterings in Pajek format. [File>Save Data As>UCINET](#) lets you save the basic network information for binary or valued networks (UCINET needs to know which) and attributes (which are stored in a separate file in UCINET).

The whole point of making more interesting drawings of graphs, of course, is to be able to use them to illustrate your ideas. There are several possibilities.

Screen capture programs (I used SnagIT) can take pictures of your graphics that can be then saved in any number of formats, and edited further by external graphics editors (perhaps to add titles, annotations, and other highlights).

[File>print](#), of course, does just that.

[File>Save Diagram](#) As let's you save your diagram in three of the most common graphics formats (Windows

Metafile, bitmap BMP, or JPEG). Once saved, these file formats can be further edited with graphics editing programs to be inserted into web or hard-copy documents.

[table of contents of this page](#)

Conclusions

A lot of the work that we do with social networks is primarily descriptive and/or exploratory, rather than confirmatory hypothesis testing. Using some of the tools described in this chapter can be particularly helpful because they may let you see patterns that you might not otherwise have seen. The tools can be used to explore tentative empirical generalizations and provide crude first examinations of hypotheses about patterns that may be present in the data.

Some of the tools are also very helpful for dealing with the complexity of social network data, which may involve many actors, many ties, and several types of ties. Hiding, highlighting, and locating parts of the data can be a big help in making sense of the data. In some cases (like ego networks and the evolution of networks over time) hiding and revealing parts of the data are critical to understanding and describing the construction and evolution of the social structures.

Finally, working with drawings can be a lot of fun, and a bit of an outlet for your creative side. A really good graphic can also be far more effective in sharing your insights than any number of words.

[table of contents of this page](#)

[table of contents of the textbook](#)

Introduction to social network methods

5. Using matrices to represent social relations

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

Contents of this chapter:

- [What is a matrix?](#)
- [The "adjacency" matrix](#)
- [Matrix permutation, blocks, and images](#)
- [Doing mathematical operations on matrices](#)
- [Transposing a matrix](#)
 - [Taking the inverse of a matrix](#)
 - [Matrix addition and matrix subtraction](#)
 - [Matrix multiplication and Boolean matrix multiplication](#)
- [Summary](#)
- [Study questions](#)

Introduction

Graphs are very useful ways of presenting information about social networks. However, when there are many actors and/or many kinds of relations, they can become so visually complicated that it is very difficult to see patterns. It is also possible to represent information about social networks in the form of matrices. Representing the information in this way also allows the application of mathematical and computer tools to summarize and find patterns. Social network analysts use matrices in a number of different ways. So, understanding a few basic things about matrices from mathematics is necessary. We'll go over just a few basics here that cover most of what you need to know to understand what social network analysts are doing. For those who want to know more, there are a number of good introductory books on matrix algebra for social scientists.

[table of contents](#)

What is a matrix?

To start with, a matrix is nothing more than a rectangular arrangement of a set of elements (actually, it's a bit more complicated than that, but we will return to matrices of more than two dimensions in a little bit). Rectangles have sizes that are described by the number of rows of elements and columns of elements that they contain. A "3 by 6" matrix has three rows and six columns; an "I by j" matrix has I rows and j columns. A matrix that has only one row is called a "row vector." A matrix that has only one column is called a "column vector."

Figure 5.1 shows a two-by-four matrix. Figure 5.2 shows a four by two matrix. Just for the moment, ignore the contents of the cells (e.g. 1,1).

Figure 5.1. Example of a "two-by-four" matrix



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

Figure 5.2. Example of at "four-by-two" matrix

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

The elements (cells) of a matrix are identified by their "addresses." Element 1,1 is the entry in the first row and first column; element 13,2 is in the 13th row and is the second element of that row. The cell addresses have been entered as matrix elements in the two examples above.

Matrices are often represented as arrays of elements surrounded by vertical lines at their left and right, or square brackets at the left and right. In web pages it's easier to use "tables" to represent matrices. Matrices can be given names; these names are usually presented as capital bold-faced letters. Social scientists using matrices to represent social networks often dispense with the mathematical conventions, and simply show their data as an array of labeled rows and columns. The labels are not really part of the matrix, but are simply for clarity of presentation. The matrix in figure 5.3 for example, is a 4 by 4 matrix, with additional labels.

Figure 5.3. Four-by-four matrix with additional row and column labels

	A	B	C	D
A	---	1	0	0
B	1	---	1	0
C	1	1	---	1
D	0	0	1	---

The matrices used in social network analysis are frequently "square." That is, they contain the same number of rows and columns. But "rectangular" matrices are also used, as are row and column vectors. The same conventions apply to all these variations.

Occasionally, social network analysts will use a "3-dimensional" matrix. A three dimensional matrix has rows, columns, and "levels" or "slices." Each "slice" has the same rows and columns as each other slice. UCINET thinks about these more complicated 3-dimensional arrays of data as a collection of two-dimensional matrices.

[table of contents](#)

The "adjacency" matrix

The most common form of matrix in social network analysis is a very simple square matrix with as many rows and columns as there are actors in our data set. The "elements" or scores in the cells of the matrix record information about the ties between each pair of actors.

The simplest and most common matrix is binary. That is, if a tie is present, a one is entered in a cell; if there

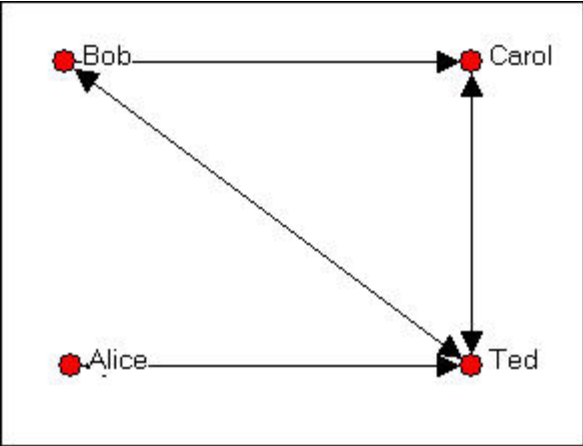
is no tie, a zero is entered. This kind of a matrix is the starting point for almost all network analysis, and is called an "adjacency matrix" because it represents who is next to, or adjacent to whom in the "social space" mapped by the relations that we have measured.

An adjacency matrix may be "symmetric" or "asymmetric." Social distance can be either symmetric or asymmetric. If Bob and Carol are "friends" they share a "bonded tie" and the entry in the $X_{i,j}$ cell will be the same as the entry in the $X_{j,i}$ cell.

But social distance can be a funny (non-Euclidean) thing. Bob may feel close to Carol, but Carol may not feel the same way about Bob. In this case, the element showing Bob's relationship to Carol would be scored "1," while the element showing Carol's relation to Bob would be scored "0." That is, in an "asymmetric" matrix, $X_{i,j}$ is not necessarily equal to $X_{j,i}$.

By convention, in a directed (i.e. asymmetric) matrix, the sender of a tie is the row and the target of the tie is the column. Let's look at a simple example. The directed graph of friendship choices among Bob, Carol, Ted, and Alice is shown in figure 5.4.

Figure 5.4 Bob, Carol, Ted, and Alice



We can since the ties are measured at the nominal level (that is, the data are binary choice data), we can represent the same information in a matrix that looks like:

Figure 5.5. Asymmetric adjacency matrix of the graph shown in figure 5.4.

	Bob	Carol	Ted	Alice
Bob	---	1	1	0
Carol	0	---	1	0
Ted	1	1	---	1
Alice	0	0	1	---

Remember that the rows represent the source of directed ties, and the columns the targets; Bob chooses Carol here, but Carol does not choose Bob. This is an example of an "asymmetric" matrix that represents directed ties (ties that go from a source to a receiver). That is, the element i,j does not necessarily equal the element j,i . If the ties that we were representing in our matrix were "bonded-ties" (for example, ties representing the relation "is a business partner of" or "co-occurrence or co-presence," (e.g. where ties represent a relation like: "serves on the same board of directors as") the matrix would necessarily be symmetric; that is element i,j would be equal to element j,i .

Binary choice data are usually represented with zeros and ones, indicating the presence or absence of each logically possible relationship between pairs of actors.

Signed graphs are represented in matrix form (usually) with -1, 0, and +1 to indicate negative relations, no or neutral relations, and positive relations. "Signed" graphs are actually a specialized version of an ordinal relation.

When ties are measured at the ordinal or interval level, the numeric magnitude of the measured tie is entered as the element of the matrix. As we discussed earlier, other forms of data are possible (multi-category nominal, ordinal with more than three ranks, full-rank order nominal). These other forms, however, are rarely used in sociological studies, and we won't give them very much attention.

In representing social network data as matrices, the question always arises: what do I do with the elements of the matrix where $i = j$? That is, for example, does Bob regard himself as a close friend of Bob? This part of the matrix is called the *main diagonal*. Sometimes the value of the main diagonal is meaningless, and it is ignored (and left blank or filled with zeros or ones). Sometimes, however, the main diagonal can be very important, and can take on meaningful values. This is particularly true when the rows and columns of our matrix are "super-nodes" or "blocks." More on that in a minute.

It is often convenient to refer to certain parts of a matrix using shorthand terminology. If I take all of the elements of a row (e.g. who Bob chose as friends: ---,1,1,0) I am examining the "*row vector*" for Bob. If I look only at who chose Bob as a friend (the first column, or ---,0,1,0), I am examining the "*column vector*" for Bob. It is sometimes useful to perform certain operations on row or column vectors. For example, if I summed the elements of the column vectors in this example, I would be measuring how "popular" each node was (in terms of how often they were the target of a directed friendship tie). So a "vector" can be an entire matrix (1 x ... or ...x 1), or a part of a larger matrix.

return to the [table of contents of this page](#)

Matrix permutation, blocks, and images

It is also helpful, sometimes, to rearrange the rows and columns of a matrix so that we can see patterns more clearly. Shifting rows and columns (if you want to rearrange the rows, you must rearrange the columns in the same way, or the matrix won't make sense for most operations) is called "permutation" of the matrix.

Our original data look like figure 5.6:

Figure 5.6. Asymmetric adjacency matrix

	Bob	Carol	Ted	Alice
Bob	---	1	1	0
Carol	0	---	1	0
Ted	1	1	---	1
Alice	0	0	1	---

Let's rearrange (permute) this so that the two males and the two females are adjacent in the matrix. *Matrix permutation* ([Data>Permute](#)) simply means to change the order of the rows and columns. Since the matrix is symmetric, if I change the position of a row, I must also change the position of the corresponding column. The result is shown in figure 5.7.

Figure 5.7. Permuted matrix

	Bob	Ted	Carol	Alice
Bob	---	1	1	0
Ted	1	---	1	1
Carol	0	1	---	0
Alice	0	1	0	---

None of the elements have had their values changed by this operation or rearranging the rows and columns, we have just shifted things around. We've also highlighted some sections of the matrix. Each colored section is referred to as a *block*. Blocks are formed by passing dividing lines through the matrix (e.g. between Ted and Carol) rows and columns. Passing these dividing lines through the matrix is called *partitioning the matrix*. Here we have partitioned by the actor by their sex. Partitioning is also sometimes called "blocking the matrix," because partitioning produces blocks.

This kind of grouping of cells is often done in network analysis to understand how some sets of actors are "embedded" in social roles or in larger entities. Here, for example, we can see that all occupants of the social role "male" choose each other as friends; no females choose each other as friends, and that males are more likely to choose females (3 out of 4 possibilities are selected) than females are to choose males (only 2 out of 4 possible choices). We have grouped the males together to create a "partition" or "super-node" or "social role" or "block." We often partition social network matrices in this way to identify and test ideas about how actors are "embedded" in social roles or other "contexts."

We might wish to dispense with the individual nodes altogether, and examine only the positions or roles. If we calculate the proportion of all ties within a block that are present, we can create a *block density matrix*. In doing this, we have ignored self-ties in figure 5.8.

Figure 5.8. Block density matrix

	Male	Female
Male	1.00	0.75
Female	0.50	0.00

We may wish to summarize the information still further by using *block image* or *image matrix*. If the density in a block is greater than some amount (we often use the average density for the whole matrix as a cut-off score, in the current example the density is .58), we enter a "1" in a cell of the blocked matrix, and a "0" otherwise. This kind of simplification is called the "image" of the blocked matrix, as in figure 5.9.

Figure 5.9. Image matrix of sex blocked data, using overall mean density as the cut-off

	Male	Female
Male	1	1
Female	0	0

Images of blocked matrices are powerful tools for simplifying the presentation of complex patterns of data. Like any simplifying procedure, good judgment must be used in deciding how to block and what cut-offs to use to create images -- or we may lose important information.

UCINET includes tools that make permuting and blocking matrices rather easy.

Transform>Block allows you to select a matrix to be blocked, a row and/or column partition, and a method for calculating the entries in the resulting blocks.

To use this command, you need to first create separate files that describe the row partition and the column partition. These files are simply vectors (either one row, or one column) that identify which actors are to fall into which partition. For example, if actors 1, 2, and 5 were to form group A, and actors 3 and 4 were to form group B, my column partition data set would read: 1 1 2 2 1. These partitions or blockings are simply regular UCINET data files with one row or one column.

The command asks for a method of summarizing the information within each block. You may take the average of the values in the block (if the data are binary, taking the average is the same thing as calculating the density), sum the values in the block, select the highest value or the lowest value, or select a measure of the amount of variation among the scores in the block -- either the sums of squares or the standard deviation.

The command outputs two new matrices. The "PrelImage" data set contains the original scores, but permuted; the "Reduced image dataset" contains a new block matrix containing the block densities.

Transform>Collapse allows you to combine rows and/or columns by specifying (detailed instructions are given on the command window) which elements are to be combined, and how. We might select, for example, to combine columns 1, 2, and 5, and rows 1, 2, and 5 by taking the average of the values (we could also select the maximum, minimum, or sum). The command creates a new matrix that has collapsed the desired rows or columns using the summary operation you selected.

The data menu also gives you some tools for this kind of work:

Data>Permute allows you to re-arrange the rows and/or columns and/or matrices (if your data set contains multiple matrices representing multiple relations, like the Knoke bureaucracies "information" and "money" relations). You simply specify the new order with a list. If I wanted to group rows 1, 2, and 5 to be new rows 1, 2, and 3; and rows 3 and 4 to be new rows 4 and 5, I would enter 1 2 4 5 3.

Data>Sort re-arranges the rows, columns, or both of the matrix according to a criterion you select. If you data are valued (i.e. represent tie strength) you might want to sort the rows and columns in ascending or descending order (this could make sense for binary data, too). If you want a more complicated sort (say "all the 3's first, then all the 1's, then all the 2's) you can use an external UCINET data file to specify this as a vector (i.e. the data set would just be: 3 1 2).

Data>Transpose re-arranges the data in a way that is very commonly used in matrix algebra -- by taking the "transpose." A transpose is, very simply, switching the rows and columns of a matrix for one another.

[table of contents](#)

Doing mathematical operations on matrices

Representing the ties among actors as matrices can help us to see patterns by performing simple manipulations like summing row vectors or partitioning the matrix into blocks. Social network analysts use a number of other mathematical operations that can be performed on matrices for a variety of purposes (matrix addition and subtraction, transposes, inverses, matrix multiplication, and some other more exotic stuff like determinants and eigenvalues). Without trying to teach you matrix algebra, it is useful to know at least a little bit about some of these mathematical operations, and what they are used for in social network analysis.

UCINET has built-in functions for doing most matrix algebra functions. Look under the **Tools>Matrix Algebra**

menu. If you do know some matrix algebra, you will find that this tool lets you do almost anything to matrix data that you may desire. But, you do need to know what you are doing. The help screen for this command shows how to identify the matrix or matrices that are to be manipulated, and the algorithms that can be applied.

Transposing a matrix

This simply means to exchange the rows and columns so that i becomes j , and *vice versa*. If we take the transpose of a directed adjacency matrix and examine its row vectors (you should know all this jargon by now!), we are looking at the sources of ties directed at an actor. The degree of similarity between an adjacency matrix and the transpose of that matrix is one way of summarizing the degree of symmetry in the pattern of relations among actors. That is, the correlation between an adjacency matrix and the transpose of that matrix is a measure of the degree of reciprocity of ties (think about that assertion a bit). Reciprocity of ties can be a very important property of a social structure because it relates to both the balance and to the degree and form of hierarchy in a network. This command is also available as [Data>Transpose](#).

Taking the inverse of a matrix

This is a mathematical operation that finds a matrix which, when multiplied by the original matrix, yields a new matrix with ones in the main diagonal and zeros elsewhere (which is called an identity matrix). Without going any further into this, you can think of the inverse of a matrix as being sort of the "opposite of" the original matrix. Matrix inverses are used mostly in calculating other things in social network analysis. They are sometimes interesting to study in themselves, however. It is sort of like looking at black lettering on white paper versus white lettering on black paper: sometimes you see different things. Inverses are calculated with [Tools>Matrix Algebra](#).

Matrix addition and matrix subtraction

These are the easiest of matrix mathematical operations. One simply adds together or subtracts each corresponding i, j element of the two (or more) matrices. Of course, the matrices that this is being done to have to have the same numbers of i and j elements (this is called "conformable" to addition and subtraction) - and, the values of i and j have to be in the same order in each matrix.

Matrix addition and subtraction are most often used in network analysis when we are trying to simplify or reduce the complexity of multiplex (multiple relations recorded as separate matrices or slices) data to simpler forms. If I had a symmetric matrix that represented the tie "exchanges money" and another that represented the relation "exchanges goods" I could add the two matrices to indicate the intensity of the exchange relationship. Pairs with a score of zero would have no relationship, those with a "1" would be involved in either barter or commodity exchange, and those with a "2" would have both barter and commodity exchange relations. If I subtracted the "goods" exchange matrix from the "money exchange" matrix, a score of -1 would indicate pairs with a barter relationship; a score of zero would indicate either no relationship or a barter and commodity tie; a score of +1 would indicate pairs with only a commodified exchange relationship. For different research questions, either or both approaches might be useful. [Tools>Matrix Algebra](#) are one way of doing these sorts of data transformations.

Matrix multiplication and Boolean matrix multiplication

Matrix multiplication is a somewhat unusual operation, but can be very useful for the network analyst. You will have to be a bit patient here. First we need to show you how to do matrix multiplication and a few important results (like what happens when you multiply an adjacency matrix times itself, or raise it to a power). Then, we will try to explain why this is useful.

To multiply two matrices, they must be "conformable" to multiplication. This means that the number of rows in the first matrix must equal the number of columns in the second. Usually network analysis uses adjacency matrices, which are square, and hence, conformable for multiplication. Multiplying a matrix by itself (i.e. raising it to a power) and multiplying a square matrix by its transpose are obviously "conformable." Unlike regular multiplication of individual numbers $X*Y$ is not the same thing as $Y*X$ in matrix multiplication -- the order matters!

To multiply two matrices, begin in the upper left hand corner of the first matrix, and multiply every cell in the first row of the first matrix by the values in each cell of the first column of the second matrix, and sum the results. Proceed through each cell in each row in the first matrix, multiplying by the column in the second. To perform a Boolean matrix multiplication, proceed in the same fashion, but enter a zero in the cell if the multiplication product is zero, and one if it is not zero. An example helps. Suppose we wanted to multiply the two matrices in figure 5.10.

Figure 5.10. Two matrices to be multiplied.

0	1
2	3
4	5

times

6	7	8
9	10	11

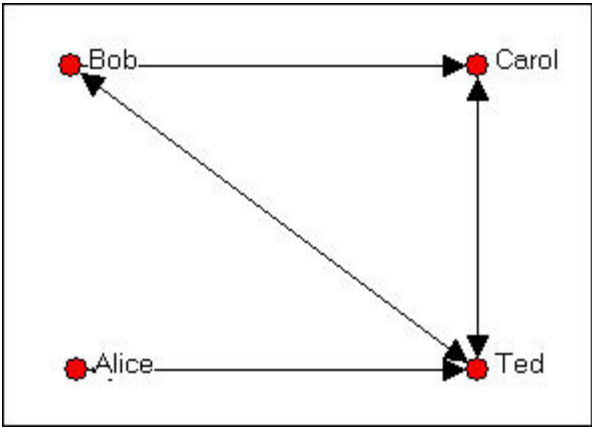
The result is shown in figure 5.11.

Figure 5.11. Result of matrix multiplication.

$(0*6)+(1*9)$	$(0*7)+(1*10)$	$(0*8)+(1*11)$
$(2*6)+(3*9)$	$(2*7)+(3*10)$	$(2*8)+(3*11)$
$(4*6)+(5*9)$	$(4*7)+(5*10)$	$(4*8)+(5*11)$

The mathematical operation in itself doesn't interest us here (any number of programs can perform matrix multiplication). But, the operation is useful when applied to an adjacency matrix. Consider our four friends again, in figure 5.12.

Figure 5.12. Directed graph of friendship relations among Bob, Carol, Ted, and Alice



The adjacency matrix for the four actors B, C, T, and A (in that order) is shown as figure 5.13.

Figure 5.13. Adjacency matrix for graph in figure 5.12.

---	1	1	0
0	---	1	0
1	1	---	1
0	0	1	---

Another way of thinking about this matrix is to notice that it tells us whether there is a path from each actor to each actor. A one represents the presence of a path, a zero represents the lack of a path. The adjacency matrix is exactly what its name suggests -- it tells us which actors are adjacent, or have a direct path from one to the other.

Now suppose that we multiply this adjacency matrix times itself (i.e. raise the matrix to the 2nd power, or square it). We will treat "self-ties" as zeros, which, effectively, ignores them. The calculation of the matrix squared is shown as figure 5.14.

Figure 5.14. Squaring matrix 5.13.

$(0*0)+(1*0)+(1*1)+(0*0)$	$(0*1)+(1*0)+(1*1)+(0*0)$	$(0*1)+(1*1)+(1*0)+(0*1)$	$(0*0)+(1*0)+(1*1)+(0*0)$
$(0*0)+(0*0)+(1*1)+(0*0)$	$(0*1)+(0*0)+(1*1)+(0*0)$	$(0*1)+(0*1)+(1*0)+(0*1)$	$(0*0)+(0*0)+(1*1)+(0*0)$
$(1*0)+(1*0)+(0*1)+(1*0)$	$(1*1)+(1*0)+(0*1)+(1*0)$	$(1*1)+(1*1)+(0*0)+(1*1)$	$(1*0)+(1*0)+(0*1)+(1*0)$
$(0*0)+(0*0)+(1*1)+(0*0)$	$(0*1)+(0*0)+(1*1)+(0*0)$	$(0*1)+(0*1)+(1*0)+(0*1)$	$(0*0)+(0*0)+(1*1)+(0*0)$

equals:

1	1	1	1
1	1	0	1
0	1	3	0
1	1	0	1

This matrix (i.e. the adjacency matrix squared) counts the number of pathways between two nodes that are of length two. Stop for a minute and verify this assertion (go back to the graph and find the paths). For example, note that actor "B" is connected to each of the other actors by a pathway of length two; and that there is no more than one such pathway to any other actor. Actor T is connected to himself by pathways of

length two, three times. This is because actor T has reciprocal ties with each of the other three actors. There is no pathway of length two from T to B (although there is a pathway of length one).

So, the adjacency matrix tells us how many paths of length one are there from each actor to each other actor. The adjacency matrix squared tells us how many pathways of length two are there from each actor to each other actor. It is true (but we won't show it to you) that the adjacency matrix cubed counts the number of pathways of length three from each actor to each other actor. And so on...

If we calculated the Boolean product, rather than the simple matrix product, the adjacency matrix squared would tell us whether there was a path of length two between two actors (not how many such paths there were). If we took the Boolean squared matrix and multiplied it by the adjacency matrix using Boolean multiplication, the result would tell us which actors were connected by one or more pathways of length three. And so on...

Now, finally: why should you care?

Some of the most fundamental properties of a social network have to do with how connected the actors are to one another. Networks that have few or weak connections, or where some actors are connected only by pathways of great length may display low solidarity, a tendency to fall apart, slow response to stimuli, and the like. Networks that have more and stronger connections with shorter paths among actors may be more robust and more able to respond quickly and effectively. Measuring the number and lengths of pathways among the actors in a network allow us to index these important tendencies of whole networks.

Individual actor's positions in networks are also usefully described by the numbers and lengths of pathways that they have to other actors. Actors who have many pathways to other actors may be more influential with regard to them. Actors who have short pathways to more other actors may be more influential or central figures. So, the number and lengths of pathways in a network are very important to understanding both individual's constraints and opportunities, and for understanding the behavior and potentials of the network as a whole.

There are many measures of individual position and overall network structure that are based on whether there are pathways of given lengths between actors, the length of the shortest pathway between two actors, and the numbers of pathways between actors. Indeed, most of the basic measures of networks, measures of centrality and power, and measures of network groupings and substructures are based on looking at the numbers and lengths of pathways among actors.

For most analyses, you won't have to manipulate matrices -- UCINET and other programs have already built algorithms that have the computer do these operations. Most of the computational work in network analysis is done with matrix mathematics though, so in order to understand what is going on, it's useful to understand the basics.

[table of contents](#)

Summary

Matrices are collections of elements into rows and columns. They are often used in network analysis to represent the adjacency of each actor to each other actor in a network. An adjacency matrix is a square actor-by-actor ($i=j$) matrix where the presence of pair wise ties are recorded as elements. The main diagonal, or "self-tie" of an adjacency matrix is often ignored in network analysis.

Sociograms, or graphs of networks can be represented in matrix form, and mathematical operations can then be performed to summarize the information in the graph. Vector operations, blocking and partitioning, and

matrix mathematics (inverses, transposes, addition, subtraction, multiplication and Boolean multiplication), are mathematical operations that are sometimes helpful to let us see certain things about the patterns of ties in social networks.

Social network data are often multiplex (i.e. there are multiple kinds of ties among the actors). Such data are represented as a series of matrices of the same dimension with the actors in the same position in each matrix. Many of the same tools that we can use for working with a single matrix (matrix addition and correlation, blocking, etc.) Are helpful for trying to summarize and see the patterns in multiplex data.

Once a pattern of social relations or ties among a set of actors has been represented in a formal way (graphs or matrices), we can define some important ideas about social structure in quite precise ways using mathematics for the definitions. In the remainder of the book, we will look at how social network analysts have formally translated some of the core concepts that social scientists use to describe social structures.

[table of contents](#)

Review questions

1. A matrix is "3 by 2." How many columns does it have? How many rows?
2. Adjacency matrices are "square" matrices. Why?
3. There is a "1" in cell 3,2 of an adjacency matrix representing a sociogram. What does this tell us?
4. What does it mean to "permute" a matrix, and to "block" it?

Application questions

1. Think of the readings from the first part of the course. Did any studies present matrices? If they did, what kinds of matrices were they (that is, what is the technical description of the kind of graph or matrix). Pick one article, and show what the data would look like, if represented in matrix form.
2. Think of some small group of which you are a member (maybe a club, or a set of friends, or people living in the same apartment complex, etc.). What kinds of relations among them might tell us something about the social structures in this population? Try preparing a matrix to represent one of the kinds of relations you chose. Can you extend this matrix to also describe a second kind of relation? (e.g. one might start with "who likes whom?" and add "who spends a lot of time with whom?").
3. Using the matrices you created in the previous question, does it make sense to leave the diagonal "blank," or not, in your case? Try permuting your matrix, and blocking it.
4. Can you make an adjacency matrix to represent the "star" network? what about the "line" and "circle." Look at the ones and zeros in these matrices -- sometimes we can recognize the presence of certain kinds of social relations by these "digital" representations. What does a strict hierarchy look like? What does a population that is segregated into two groups look like?

[table of contents](#)

[table of contents of the book](#)

Introduction to social network methods

6. Working with network data

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

Contents of this chapter

- [Introduction: Manipulating network data structures](#)
- [Making UCINET datasets](#)
- [Transforming data values](#)
- [File handling tools](#)
- [Selecting sub-sets of the data](#)
- [Making new kinds of graphs from existing graphs](#)
- [Conclusion](#)

Introduction: Manipulating network data structures

This chapter is about the kinds of "data structures" that network analysts work with most frequently, and some of the most common kinds of transformations and manipulations of these structures.

Data Structures

Most everyone reading this is very familiar with the kind of "data structure" that is used in many statistical studies. The rectangular array of data that we are used to seeing in SPSS, SAS, Excel, and other programs is a "structure" that is defined by its rows (which represent cases) and columns (which represent variables). An example is shown as figure 6.1.

Figure 6.1. Rectangular data array

ID	Sex	Age	Married
Bob	M	42	1
Carol	F	44	1
Ted	M	39	0
Alice	F	27	0

Earlier, we emphasized that the social network perspective leads us to focus our attention on the relations between actors, more than on the attributes of actors. This approach often results in data that have a different "structure" in which both rows and columns refer to the same actors, and the cells report information on one variable that describes variation (in the case of the example below, simple presence of absence of a tie) in the relations between each pair of actors. An example is given as figure 6.2.

Figure 6.2. Square data structure for social network data

Friendship ties				

	Bob	Carol	Ted	Alice
Bob	---	1	0	0
Carol	0	---	1	0
Ted	1	1	---	1
Alice	0	0	1	---

A "data structure" is simply the way in which information is recorded. These two examples are both two-dimensional (rows and columns). It is possible, for a data structure or data object to have more than two dimensions. For example, if we wanted to also record information about the network relations of who is married to whom, we would usually create another table of actors by actors (that is, the row and column indexes would be the same), and record the presence or absence of marital ties. If we "stacked" the two tables together, we would have a 4 by 4 by 2 "data structure." Counts of the rows, columns, and matrices (or "slices") do not include the labeling or indexing information (i.e. it's not 5 x 5 x 3).

Social network analysis data structures:

Network analysts work with a variety of data structures. In this chapter, we'll look tools for creating and manipulating the most common types.

One major "type" of data structure is the actor-by-actor matrix (like the friendship data above). This kind of structure is, by definition, a "two-dimensional," and "square" (the number of rows and columns are equal). The information in each cell provides information about the relation between a particular pair of actors.

The two-dimensional actor-by-actor matrix is very often expanded into a "third dimension" by adding "slices" that represent additional kinds of relations among the actors. For example, we might have an actor-by-actor matrix of Bob, Carol, Ted, and Alice that records the degree of "liking" directed from each to each. In addition, we might add a second "slice" that records the presence or absence of a kinship relation between each pair. These kinds of 3-dimesional network data structures are "multi-plex." That is, they represent multiple relations among the same sets of actors. Some of the special issues in working with multi-plex data are discussed in chapter 15.

The other major "type" of data structure that network analysts use looks a lot like the "rectangular data array" from conventional statistical work. The data structure consists of rows (representing actors) by columns (representing attributes of each actor -- what would be called "variables" in statistics). Such an array might record just one attribute, in which case the data structure would be a "column vector." Or, such an array might record a number of attributes of each actor. Network analysts think of this kind of "rectangular" array of actors by attributes simply as a collection of vectors.

The "rectangular" data structure (called an "attribute" data set) is used in a number of ways in network analysis.

- It can record attributes of each actor that we know from other sources (e.g. gender, age, etc.).
- It can record attributes of each actor that arise from their position in the network itself (e.g. the "between-ness centrality" score of each actor).
- It can record what part or sub-part of an network an actor falls in. For example, a column in an "attribute" data structure might consist of the letters "A" "B" and "C" to indicate which of three "factions" each actor was a member of. This is called a "partition."
- It can be used to tell UCINET how the actors in a matrix are to be re-arranged, or "permuted."

The "rectangular" data structure can also be used to record information about the relationships between two types of nodes (called bi-partite data). This use is so common and so important that it has a special name --

and "incidence" or an "affiliation" matrix. For example, the rows might be indexed by actors (e.g. Bob, Carol...); but, the columns might be the organizations that employ the actors (IBM, Sun, Microsoft...). Entries in the cells indicate the presence or strength of the relation between an actor and an employer.

Incidence or affiliation data is particularly important in many social network analyses because it is "multi-level." Actors may be tied together because they are present in the same place, time, or category (that is, they are in the same "incident" to, or are "affiliated" with the same structure). But such data also show how "incidents" are tied together by the "co-presence" of actors. Incidence data involving two kinds of actors (bi-partite) data are very important in network analysis because they are often our best window into questions of "agency and structure" or "macro-micro linkages."

In this chapter we will describe some of the most common kinds of manipulations that social network analysts use in creating data structures, and changing their structures to meet the needs of particular research questions. Even though this chapter is going to be a bit long, it hardly covers all the possibilities. Different questions require different data structures. The examples and tools here will get you started.

[table of contents](#)

Making UCINET datasets

UCINET datasets are stored in a special (Pascal) format, but can be created and manipulated using both UCINET's and other software tools (text editors and spreadsheets). Each UCINET dataset consists of two separate files that contain header information (e.g. myfile.##h) and the data lines (e.g. myfile.##d). Because of this somewhat unusual way of storing data, it is best to create data sets with the internal spreadsheet editor or DL language tools, or to import text or spreadsheet files and save the results as UCINET files.

There are several ways of creating data files that UCINET can read.

The spreadsheet editor. UCINET has a built-in spreadsheet editor that can be used to enter case and variable labels and data values ([data>Spreadsheets>matrix](#)). This editor allows you to specify the number of rows and columns, and has the nice feature of being able to specify that a data set is symmetric. If we are recording a data set where ties among actors are not directed, this feature saves half the data entry. There are also tools to fill the matrix with zeros (a common starting point for many data sets that have sparse connections among actors), permuting rows, symmetrizing and dichotomizing (see discussions in the sections below), and copying row labels to the column labels (if the data are symmetric, you need only enter the labels once).

The UCINET spreadsheet editor can import and export Excel spreadsheets, so you can use tools in both programs to full advantage. To import Excel to UCINET, be sure to save your spreadsheet as version 4 or earlier; the multi-sheet format of more recent Excel versions isn't supported in UCINET.

If you have a fairly small dataset, the UCINET spreadsheet editor is a good choice for making single matrix datasets, which are automatically saved as UCINET files that can be used by other parts of the program.

Importing (and Exporting). Data sets can be moved from a number of other program's data file formats into UCINET's. The [Data>Import>...](#) menu item supports import from NetDraw (VNA format), Pajek, Krackplot, and Negopy. It also supports importing raw ASCII text files, and files saved as Excel spreadsheets (version 4 or earlier). So, if you started with a NetDraw drawing, for example, and saved the results as VNA, you may import this into UCINET for calculating network measures. I'm more comfortable with Excel than with UCINET's editor, so I usually make data sets in Excel, and import them.

When UCINET imports a file, it will produce a window with your results. Check to make sure they are

correct! When the import is performed, UCINET automatically saves the data files in UCINET format in the default directory.

It's often a good idea to set up a new directory for each project, and to set the default to this new directory using the file-cabinet icon on the toolbar, or [File>Change default](#) folder.

UCINET datasets can also be exported for use in other programs. [Data>Export>...](#) will produce Excel, raw ASCII text, Pajek, Mage, Metis, and Krackplot files.

The DL language: If you've been looking at the UCINET Data menu as you read the preceding discussion, you may have noted that the program imports and exports "DL" files. DL (for "data language") is a very powerful and (fairly) simple language that allows the creation of quite complex and large UCINET data sets with minimal data entry.

DL language files are plain ASCII text files that can be created with any editor (be sure to store the results as plain text). A quite complete reference guide is provided in UCINET ([Help>Help Topics>DL](#)).

The DL language can be a bit picky, and it does take a little effort to figure out how to do exactly what you want to do with it. But, there are a number of circumstances where it is well worth the effort -- when compared to using a spreadsheet. Particularly, if your data set consists of multiple matrices, and if the data are fairly sparse, or if the data set has many rows and columns; then the DL file is the right way to go.

We won't explore the language in any detail here -- the help file is quite good. Figure 6.3 shows an example of a DL file that illustrates a few of the features.

Figure 6.3. Example DL language file

```
dl n=9, format=edgelist1
labels:
    A,B,C,D,E,F,G,H,I
data:
    1  1  1
    1  2  1
    1  6  1
    .
    .
    8  7  1
    9  9  1
```

The file begins with "dl" to indicate file type, and specification of the dimension of the data structure (the language allows specification of number of rows, columns, and matrices). Labels for the nodes are given in the "labels:" paragraph. The data are given in a "data:" paragraph.

The interesting thing in this example is the use of the *format=edgelist1* command. This tells UCINET to read the data lines in a way that is very efficient. The edgelist1 format is a set of rows, each one of which identifies two nodes and the value of the connection between them. In the resulting data set, all entries are zero, except those that have been specified. So, among our nine actors, there is a tie from actor 1 to actor 1, a tie from actor 1 to actor 2, a tie from actor 1 to actor 6, etc. Here, the matrix is binary -- the value of each tie (the third entry on each line) is 1.

Another very useful *format=* method is *nodelist1*. In this format, each line of data consists of the name (or number) of an origin node, followed by all of the nodes to which it has a connection (this particularly format is for zero/one data on the presence or absence of a connection). This approach then requires only one line of data for each actor. For example, a line in the *data:* section that read: 3 5 6 19 24 would indicate that actor

number 3 had a binary directed tie to actors 5, 6, 19, and 24.

These, and other methods available in DL allow the entry of very large and complex data sets with the greatest efficiency and minimum typing. If you are facing a problem with many cases, connections, or kinds of connections, invest a little time in DL.

[table of contents](#)

Transforming data values

It is not at all unusual for the analyst to want to change the values that describe the relations between actors, or the values that describe the attributes of actors. Suppose the attribute "gender" had been entered into a data set using the values "1" and "2," and we wanted to change the attribute to be "Female" coded as "0" and "1." Or, suppose that we had recorded the strength of ties between companies by counting the number of members of boards of directors that they had in common. But we then decide that we are only interested in whether there are members in common or not. We need to change the coded values to be equal to "0" if there are no board members in common, and "1" if there are any (regardless of how many).

Just like statistical packages, UCINET has built-in tools that do some of the most common data transformations.

[Transform>Recode](#) is a very flexible and general purpose tool for recoding values in any UCINET data structure. its dialog box has two tabs: "[Files](#)" and "[Recode](#)."

In the *files* tab, you can browse for an input dataset, select which matrices in the set to recode (if there is more than one), which rows and columns to recode (this is good if you are working on a collection of attribute vectors, for example, and only want to recode selected ones), whether to recode the values on the diagonal, and the name of the output dataset.

In the *recode* tab, you specify what you want done by creating rules. For example, if I wanted to recode all values 1, 2, and 3 to be zero; and any values of 4, 5, and 6 to be one, I would create two rules. "Values from 1 to 3 are recoded as 0" "Values from 4 to 6 are recoded as one." The rules are created by using simple built-in tools.

Almost any transformation in a data set of any complexity can be done with this tool. But, often there are simpler ways to do certain tasks.

[Transform>Reverse](#) recodes the values of selected rows, columns, and matrices so that the highest value is now the lowest, the lowest is now the highest, and all other values are linearly transformed. For example, the vector: 1 3 3 5 6 would become the vector 6 4 4 2 1.

If we've coded a relationship as "strength of tie" but want our results to be about "weakness of tie" a "reverse" transform would be helpful.

A common task in network analysis is to calculate the "similarity" or "distance" between two actors based on their relationships with other actors (more on this in the sections on equivalence, later). "Similarity" scores can be "reversed" to become "dis-similarity;" "distance" scores can be "reversed" to be "nearness" scores with this tool.

[Transform>Dichotomize](#) is a tool that is useful for turning valued data into binary data. That is, if we have measured the strength of ties among actors (e.g. on a scale from 0 = no tie to 5 = strong tie), the "dichotomize" can be used to turn this into data that represent only the absence or presence of a tie (e.g.

zero or one).

Why would one ever want to do this? To convert an ordinal or interval measure of relation strength into simple presence/absence may throw away considerable amounts of information. Many of the tools of social network analysis were developed for use with binary data only, and give misleading results (or none at all!) when applied to valued data. Many of the tools in UCINET that are designed for binary data will arbitrarily dichotomize interval or ordinal data in ways that might not be appropriate for your problem.

So, if your data are valued, but the tool you want to use requires binary data, you can turn your data into zero-one form by selecting a cut-off value (you will also have to select a "cut-off operator" and decide what to do with the diagonal).

Suppose, for example, I'd measured tie strength on a scale from 0 to 3. I'd like to have the values 0 and 1 treated as "0" and the values 2 and 3 treated as "1." I would select "greater than" as the cut-off operator, and select a cut-off value of "2." The result would be a binary matrix of zeros (when the original scores were 0 or 1) and ones (when the original scores were 2 or 3).

This tool can be particularly helpful when examining the values of many network measures. For example, the shortest distance between two actors ("geodesic distance") might be computed and saved in a file. We might then want to look at a map or image of the data at various levels of distance -- first, only display actors who are adjacent (distance = 1), then actors who are one or two steps apart, etc. The "dichotomize" tool could be used to create the necessary matrices.

[*Transform>Diagonal*](#) lets you modify the values of the ties of actors with themselves, or the "main diagonal" of square network data sets. The dialog box for this tool allows you to specify either a single value that is to be entered for all the cells on the diagonal; or, a list of (comma separated) values for each of the diagonal cells (from actor one through the last listed actor).

For many network analyses, the values on the main diagonal are not very meaningful, and you may wish to set them all to zero or to one -- which are pretty common approaches. Many of the tools for calculating network measures in UCINET will automatically ignore the main diagonal, or ask you whether to include it or not.

On some occasions, though, you may wish to be sure that ties of an actor with themselves are treated as present (e.g. set diagonal values to 1), or treated as absent (e.g. set diagonal values to zero).

[*Transform>Symmetrize*](#) is a tool that is used to turn "directed" or "asymmetric" network data into "undirected" or "symmetric" data.

Many of the measures of network properties computed by UCINET are defined only for symmetric data (see the help screens for information about this). If you ask to calculate a measure that is defined for only symmetric data, but your data are not symmetric, UCINET either will refuse to calculate a measure, or will symmetrize the data for you.

But, there are a number of ways to symmetrize data, and you need to be sure that you choose an approach that makes sense for your particular problem. The choices that are available in the [*Transform>Symmetrize*](#) tool are:

[*>Maximum*](#) looks at each cell in the upper part of the matrix and the corresponding cell in the lower part of the matrix (e.g. cell 2, 5 is compared to cell 5, 2), and enters the larger of the values found into both cells (e.g. 2, 5 and 5, 2 will now have the same output value). For example, suppose that we felt that the strength of the tie between actor A and actor B was best represented as being the strongest of the ties between them (either A's tie to B, or B's tie to A, whichever was strongest).

>*Minimum* characterizes the strength of the symmetric tie between A and B as being the weaker of the ties AB or BA. This corresponds to the "weakest link," and is a pretty common choice.

>*Average* characterizes the strength of the symmetric tie between A and B as the simple average of the ties AB and BA. Honestly, I have trouble thinking of a case where this approach makes a lot of sense for social relations.

>*Sum* characterizes the strength of the symmetric tie between A and B as the sum of AB and BA. This does make some sense -- that all the tie strength be included, regardless of direction.

>*Difference* characterizes the strength of the symmetric tie between A and B as $|AB - BA|$. So, relationships that are completely reciprocal end up with a value of zero; those what are completely asymmetric end up with a value equal to the stronger relation.

>*Product* characterizes the strength of the symmetric relation between A and B as the product of AB and BA. If reciprocity is necessary for us to regard a relationship as being "strong" then either "sum" or "product" might be a logical approach to symmetrizing.

>*Division* characterizes the strength of the symmetric relation between A and B as AB/BA . This approach "penalizes" relations that are equally reciprocated, and "rewards" relations that are strong in one direction, but not the other.

>*Lower Half* or >*Upper Half* uses the values in one half of the matrix for the other half. For example, the value of BA is set equal to whatever AB is. This transformation, though it may seem odd at first, is quite helpful. If we are interested in focusing on the network properties of "senders" we would choose to set the lower half equal to the upper half (i.e. select Upper Half). If we were interested in the structure of tie receiving, we would set the upper half equal to the lower.

>*Upper > Lower* or >*Upper < Lower* (and similar functions available in the dialog box) compare the values in cell AB and BA, and return one or the other based on the test function. If, for example, we had selected Upper > Lower and $AB = 3$ and $BA = 5$, the function would select the value "5," because the upper value (AB) was not greater than the lower value (BA).

>*Transform>Normalize* provides a number of tools for rescaling the scores in rows, in columns, or in both dimensions of a matrix of valued relations. A simple example might be helpful.

Figure 6.4 shows some data (from the United Nations Commodity Trade database) on trade flows, valued in dollars, of softwood lumber among 5 major Pacific Rim nations at c. 2000.

Figure 6.4. Value of softwood lumber exports among five nations


```

IMPORT FROM EXCEL
-----
Input Excel file      C:\Documents and Settings\hanneman\My
Output UCINET dataset: C:\Documents and Settings\hanneman\My

Lumber_trade

      I

      1      2      3      4      5
      Canada China Japan Mexico USA
-----
1 Canada      0      7676951 1153248512      277121 6203852800
2 China      34647      0      32261908      0      72341
3 Japan      0      457308      0      0      239941
4 Mexico      0      12481      27410      0      25357048
5 USA      103262528 21090696 81825120 61437484      0

-----
Running time: 00:00:06
Output generated: 26 Jan 05 08:48:09
Copyright (c) 1999-2004 Analytic Technologies

```

Suppose we were interested in exploring the structure of export partner dependence -- the disadvantage that a nation might face in establishing high prices when it has few alternative places to sell its products. For this purpose, we might choose to "normalize" the data by expressing it as row percentages. That is, what proportion of Canada's exports went to China, Japan, etc. Using the row normalization routine, we produce figure 6.5.

Figure 6.5. Row (sending or export) normalized lumber trade data

```

NORMALIZE
-----
Dimension:      Rows
Method:         Marginal
Diagonal valid? NO
Input dataset:  C:\Documents and Settings\hanneman\My

      1      2      3      4      5
      Canad China Japan Mexic  USA
-----
1 Canada      0.001 0.157 0.000 0.842
2 China      0.001 0.997 0.000 0.002
3 Japan      0.000 0.656 0.000 0.344
4 Mexico      0.000 0.000 0.001 0.998
5 USA      0.386 0.079 0.306 0.230

Normalized matrix saved as dataset C:\Documents and Settings\hanneman\My

```

Graphing the original trade-flow data would answer our question, but graphing the row normalized data gives us a much clearer picture of export dependency. If we were interested in import partner trading concentration, we might normalize the data by columns, producing figure 6.6.

Figure 6.6. Column (receiving or import) normalized lumber trade data

NORMALIZE						
Dimension:		I	Columns			
Method:			Marginal			
Diagonal valid?			NO			
Input dataset:			C:\Documents and S			
		1	2	3	4	5
		Canad	China	Japan	Mexic	USA
1	Canada		0.263	0.910	0.004	0.996
2	China	0.000		0.025	0.000	0.000
3	Japan	0.000	0.016		0.000	0.000
4	Mexico	0.000	0.000	0.000		0.004
5	USA	1.000	0.721	0.065	0.996	
Normalized matrix saved as dataset C:\Documents						

We see, for example, that all of Canada's imports are from the USA, and that virtually all of the USA's imports are from Canada.

The [>Transform>Normalize](#) tool provides a number of ways of re-scaling the data that are frequently used.

Normalization may be applied to either rows or columns (as in our examples, above), or it may be applied to the entire matrix (for example, rescaling all trade flows as percentages of the total amount of trade flow in the whole matrix). Normalization may also be applied to both rows and columns, iteratively. For example, if we wanted an "average" number to put in each cell of the trade flow matrix, so that both the rows and the columns summed to 100%, we could apply the iterative row and column approach. This is sometimes used when we want to give "equal weight" to each node, and to take into account both outflow (row) and inflow (column) structure.

There are a number of alternative, commonly used, approaches to how to rescale the data. Our examples use the "marginal" total (row or column) and set the sum of the entries in each row (or column) to equal 1.0. Alternatively, we might want to express each entry relative to the mean score (e.g. divide each element of the row by the average of the elements in a row). Alternatively, one might rescale by dividing by the standard deviation, or both mean and standard deviation (i.e. express the elements as Z scores). UCINET supports all of these as built-in functions. In addition, scores can be normalized by Euclidean norm, or by expressing each element as a percentage of the maximum element in a row.

Rescaling transforms like these can be very, very helpful in highlighting structural features of the data. But, obviously different normalizing approaches highlight very different features. Try thinking through how what applying each of the available transformations would tell you for some data that describe a relation that you are interested in. Some of the transformations will be completely useless; some may give you some new ideas.

[table of contents](#)

File handling tools

Because UCINET data files are stored in a somewhat unusual dual-file format, it is usually most convenient to do basic file-handling tasks within UCINET. The program has basic file handling tools within it. Using these has the advantage of automatically dealing with both the .##h and .##d files that make up each UCINET dataset. If you use file handling commands outside UCINET (e.g. using Windows), you need to remember to deal with both files for each data set.

File utilities:

File>Copy UCINET Dataset

File>Rename UCINET Dataset

File>Delete UCINET Dataset

These commands do exactly what usual operating system commands do, but manage both component files with a single command.

Viewing the contents of files:

Data>Browse is a tool for examining parts of a dataset. You select the dataset, and specify which rows, columns, and labels you would like to see. This can be very useful if the dataset you're working with is a large one, and your interest is in examining only a portion of it.

Data>Display also allows you to modify how you see a data file. You may set field width, numbers of decimals to display, whether to show zeros or not; in addition, you can select which rows and/or columns to display (the row and column numbers are specified as comma delimited lists, and can use "AND" and "OR"). If the data have been grouped into "blocks," and the block memberships have been stored as UCINET datasets, these may be used to present the data with members of blocks adjacent to one another.

Data>Describe provides basic information about a file (numbers of rows, columns, matrices). It also shows labels, and allows you import row and column labels from an external text file (just prepare an ASCII text file with the labels in rows, or comma delimited). You can also use this little utility to add a longer descriptive title to a UCINET data set. This is often a good idea if you are working with a number of related data sets with similar names.

[table of contents](#)

Selecting sub-sets of the data

As we work on understanding the structure of a social network, there are occasions when we may wish to focus our attention on only a portion of the actors. Sometimes it's just a matter of clearing away "underbrush" of nodes that aren't "important." Sometimes it's a matter of selecting sets of actors for separate consideration.

UCINET has a number of built-in tools that can be useful for creating new data sets from existing data sets, that include only portions of the actors.

Data>Extract is a general-purpose tool that allows you to either "keep" or to "delete" rows, columns, or matrices for output to a new dataset. You may select the rows, columns, or relations (matrices) to keep by listing them in external data files, or by choosing the names of the rows, columns or matrices from drop-down lists.

Data>Extract main component retains all the nodes and relations among nodes that are part of the largest component of a graph. In English: the information about the actor and connections among the actors who are part of the largest set of actors who are all connected is retained. If a graph contains several components (e.g. if there are some "isolates" or there are sub-groups who have no connection to the largest group) only the largest will be retained. Many analyses require that all the nodes be connected. But, not all real networks actually are. So, you may wish to extract the largest component and analyze it.

Data>Subgraphs from partitions is a (somewhat more complicated) tool that lets you divide the cases into groups (partitions), and output separate data files for each group. The first step (after you've decided which cases fall in which partition), is to create an external data file that lists partition membership. Suppose I wanted to put nodes 1, 3, and 5 in one value of a partition (i.e. in one group) and cases 2, 4, and 6 in another. I'd create a data file that looked like: 1, 2, 1, 2, 1, 2. This says, put the first node in partition one, put the second node in partition two, put the third node in partition one, etc. This filename is supplied to the *>Subgraphs from partitions* dialog. You may also limit the process by electing to output only certain partitions (list them in the dialog window), and/or to create new data sets for a partition value only if there are more than some number (which you specify) of cases.

Many network analysis algorithms generate information on partition membership (and save partition membership information as files you can plug in to this utility). You might also want to impose your own partitions to identify people in one community, people of a particular gender, etc.

Data>Remove isolates creates a new data set that contains all cases that are not isolated. An "isolate" is a case that has no connections at all to any other actors. Sometimes, when we collect information by doing a census of all the actors of a given type, or in a given location, some are "isolated." While this is usually an interesting social fact, we may wish to focus our attention on the community of actors who are connected (though not necessarily forming a single "component").

Data>Remove pendants creates a new data set that contains all cases that are not "pendants." A "pendant" is a case that is connected to the graph by only one tie; cases like these will "dangle" off of more central cases that are more heavily connected. In looking at large graphs with many actors, we may wish to limit our attention to nodes that are connected to at least two other actors -- so as to focus attention on the "core" of the network. Removing isolates and pendants can help to clear some of the "clutter."

Data>Egonet is a tool that lets us extract particular actors and those in their immediate "neighborhood" as separate datasets. As we will see later on, the "ego-network" of a single actor, or of some selection of actors (all men, all cases with high between-ness, etc.) is often the focus of investigation more than the structure of the whole network.

An "ego-network" is the set of actors who are connected to a focal actor, along with the relations between ego and the alters, and any relations among the alters. The structure of ego networks (whether they are dense or thin, and whether they contain "structural holes" are often critical variables in understanding and predicting the behavior of "ego."

The *Data>Egonet* tool lets you list the "egos" or "focal nodes" you'd like to extract by using an external file list or by selecting their labels from a drop-down list. The dialog asks whether you want to include ego, or only to retain information on ego's neighbors; the most common, and default, choice is to include ego as well as ego's neighbors.

Data>Unpack is a tool for creating a new data set that contains a sub-set of matrices from a larger data set. For example, if we had stored information on both "liking" and "spouse" relation in a single data set, we can use this utility to create separate data files for one or both relations. The relations to be "unpacked" are selected from a drop-down box.

Data>Join is a tool that can be used to combine separate sets of data into a new data set. Often we collect attribute information about actors in several different settings (e.g. several classrooms in a school) and store these as separate files. Or, we may have multiple files that contain information about different attributes of actors (for example, one file might be things we know from outside sources like age, sex, etc.; another file might contain information on which partition of a graph each actor falls into). We might want to combine all the attribute information into a single file. Or, we might have information about different relations among the

same set of actors, that have been stored as separate data files (as in the "liking" and "spouse" relations example).

Using [Data>Join>Rows](#) will combine two or more matrices (stored as separate files) into a single matrix that has rows for all nodes in each of the files. If I had separate files that listed the age of students in each of two classrooms, and I wanted to create a single file with all the students, the "rows" approach would be used.

Using [Data>Join>Columns](#) will combine two or matrices (stored as separate files) into a single matrix that has the same number of rows as each of the input files, but appends the columns. If I had information on age and sex for actors A, B, and C in one file and information on centrality and degree for actors A, B, and C in another, I could do a column join to produce a file that listed age, sex, centrality, and degree for actors A, B, and C.

Using [Data>Join>Matrices](#) will combine information on multiple relations among the same sets of actors into a single file. Each input file has the same actors by actors array, but for different relations. The output file combines the multiple files into a three-dimensional array of actor by actor by relation.

[table of contents](#)

Making new kinds of graphs from existing graphs

Turning attributes into relations

At the beginning of this chapter we looked at the "data structures" most commonly used in network analysis. One was the node-by-node square matrix, to record the relations between pairs of actors; and its more general "multi-plex" form to record multiple relations among the same set of actors. The other was the rectangular matrix. This "actor by attribute" matrix is most commonly used to record information about the variable properties of each node.

Network analysis often finds it useful to see actor attributes as actually indicating the presence, absence, or strength of "relations" among actors. Suppose two persons have the same gender. To the non-network analyst, this may represent a statistical regularity that describes the frequencies of scores on a variable. A network analyst, though, might interpret the same data a bit differently. A network analyst might, instead, say "these two persons share the relation of having the same gender."

Both interpretations are, of course, entirely reasonable. One emphasizes the attributes of individuals (here are two persons, each is a woman); one emphasizes the relation between them (here are two persons who are related by sharing the same social role).

It's often the case that network researchers, who are pre-disposed to see the world in "relational" terms, want to turn "attribute" data into "relational" data for their analyses.

[Data>Attribute](#) is a tool that creates an actor-by-actor relational matrix from the scores on a single attribute vector. Suppose that we had an attribute vector stored in a UCINET file (other vectors could also be in the file, but this algorithm operates on a single vector), that measured whether each of 100 large donors had given funds in support of (+1) or in opposition to (-1) a given ballot proposition. Those who made no contribution are coded zero.

We might like to create a new matrix that identifies pairs of actors who shared support or shared opposition to the ballot initiative, or who took opposite positions. That is, for each pair of actors, the matrix element is "1" if the actors jointly supported or jointly opposed the proposition, "-1" if one supported and the other opposed, and zero otherwise (if either or both made no contribution).

Using the [Data>Attribute](#) tool, we can form a new square (actor-by-actor) matrix from the scores of actors on one attribute in a number of ways. The [Exact Matches](#) choice will produce a "1" when two actors have exactly the same score on the attribute, and zero otherwise. The [Difference](#) choice will create a new matrix where the elements are the differences between the attribute scores of each pair of actors (alternatively, the [Absolute Difference](#), or [Squared Difference](#) choices will yield positively valued measures of the distance between the attribute scores of each pair of actors. The [Sum](#) choice yields a score for each pair that is equal to the sum of their attribute scores. In our current example, the [Product](#) choice (that is, multiply the score of actor i times the score of actor j, and enter the result) would yield a score of "1" if two actors shared either support or opposition, "-1" if they took opposed stands on the issue, or "0" if either did not take a position.

The [Data>Attribute](#) tool can be very useful for conceptually turning attributes into relations, so that their association with other relations can be studied.

[Data>Affiliations](#) extends the idea of turning attributes into relations to the case where we want to consider to multiple attributes. Probably the most common situations of this type are where the multiple "attributes" we have measured are "repeated measures" of some sort. Davis, for example, measured the presence of a number of persons (rows) at a number of parties (attributes or columns). From these data, we might be interested in the similarity of all pairs of actors (how many times were they co-present at the same event?), or how similar were the parties (how much of the attendance of each pair of parties were the same people?).

The example of donors to political campaigns can be seen in the same way. We might collect information on whether political donors (rows) had given funds against or for a number different ballot propositions (columns). From this rectangular matrix, we might be interested in forming a square actor by actor matrix (how often do each pair of actors donate to the same campaigns?); we might be interested in forming a square campaign by campaign matrix (how similar are the campaigns in terms of their constituencies?).

The [Data>Affiliations](#) algorithm begins with a rectangular (actor-by-attribute) matrix, and asks you to select whether the new matrix is to be formed by rows (i.e. actor-by-actor) or columns (i.e. attribute-by-attribute).

There are different ways in which we could form the entries of the new matrix. UCINET provides two methods: [Cross-Products](#) or [Minimums](#). These approaches produce the same result for binary data, but different results for valued data.

Let's look at the binary case first. Consider two actors "A" and "B" who have made contributions (or not) to each of 5 political campaigns, as in figure 6.7.

Figure 6.7. Donations of two donors to five political campaigns (binary data)

	Campaign 1	Campaign 2	Campaign 3	Campaign 4	Campaign 5
"A"	0	0	1	1	1
"B"	0	1	1	0	1

The [Cross-Products](#) method multiplies each of A's scores by the corresponding score for B, and then sums across the columns (if we were creating a campaign-by-campaign matrix, the logic is exactly the same, but would operate by multiplying columns, and summing across rows). Here, this results in: $(0*0) + (0*1) + (1*1) + (1*0) + (1*1) = 2$. That is, actors A and B have two instances where they both supported a campaign.

The [Minimums](#) method examines the entries of A and B for campaign 1, and selects the lowest score (zero). It then does this for the other campaigns (resulting in 0, 1, 0, 1) and sums. With binary data, the results will be the same by either method.

With valued data, the methods do not produce the same results; they get at rather different ideas.

Suppose that we had measured whether A and B supported (+1), took no position (0), or opposed (-1) each of the five campaigns. This is the simplest possible "valued" data, but the ideas hold for valued scales with wider ranges, and with all positive values, as well. Now, our data might look like those in figure 6.8.

Figure 6.8. Donations of two donors for or against five political campaigns (valued data)

	Campaign 1	Campaign 2	Campaign 3	Campaign 4	Campaign 5
"A"	-1	0	1	-1	1
"B"	-1	1	1	0	-1

Both A and B took the same position on two issues (both opposed on one, both supporting another). On two campaigns (2, 4), one took no stand. On issue number 5, the two actors took opposite positions.

The *Cross-products* method yields: $(-1 * -1) + (0 * 1) + (1 * 1) + (-1 * 0) + (1 * -1)$. That is: $1 + 0 + 1 + 0 - 1$, or 1. The two actors have a "net" agreement of 1 (they took the same position on two issues, but opposed positions on one issue).

The *Minimums* method yields: $-1 + 0 + 1 - 1 - 1$ or -2. In this example, this is difficult to interpret, but can be seen as the net number of times either member of the pair opposed an issue. The minimums method produces results that are easier to interpret when all values are positive. Suppose we re-coded the data to be: 0 = opposed, 1 = neutral, and 2 = favor. The minimums method would then produce $0 + 1 + 2 + 0 + 0 = 3$. This might be seen as the extent to which the pair of actors jointly supported the five campaigns.

Turning relations into attributes

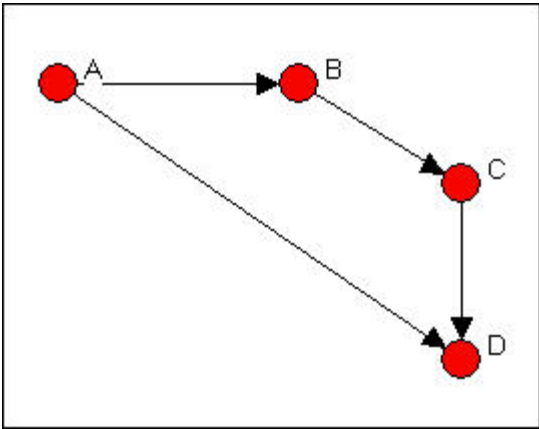
Suppose that we had a simple directed relation, represented as a matrix as in figure 6.9.

Figure 6.9. Linegraph example matrix

		1	2	3	4
	A	B	C	D	
		-	-	-	-
1	A	0	1	0	1
2	B	0	0	1	0
3	C	0	0	0	1
4	D	0	0	0	0

This is easier to see as a graph, as in figure 6.10.

Figure 6.10. Linegraph example graph



Now suppose that we are really interested in describing and thinking about the relations, and the relations among the relations -- rather than the actors, and the relations among them. That sounds odd, I realize. Let me put it a different way. We can think about the graph in Figure 6.5 as composed of four relations (A to B, B to C, C to D, and A to C). These relations are connected by having actors in common (e.g. the A to B and the B to C relations have the actor B in common). That is, we can think about relations as being "adjacent" when they share actors, just as we can think about actors being adjacent when they share relations.

Transform>Incidence is an algorithm that changes the way we look at a directed graph from "actors connected by relations" to "relations connected by actors." This is sometimes just a semantic trick. But, sometimes it's more than that -- our theory of the social structure may actually be one about which relations are connected, not which actors are connected. If we apply the *Transform>Incidence* algorithm to the data in figures 6.4 and 6.5, we get the result in figure 6.11.

Figure 6.11. Incidence matrix of graph 6.10

		1	2	3	4
		1-	1-	2-	3-
1	A	1	1	0	0
2	B	-1	0	1	0
3	C	0	0	-1	1
4	D	0	-1	0	-1

Each row is an actor. Each column is now a relation (the relations are numbered 1 through 4). A positive entry indicates that an actor is the source of a directed relation. For example, actor A is the origin of the relation "1" that connects A to B, and is a source of the relation "2" that connects actor A to actor D. A negative entry indicates that an actor is the "sink" or recipient of a directed relation. For example, actor C is the recipient in relation "3" (which connects actor B to actor C), and the source of relation "4" (which connects actor C to actor D).

The "incidence" matrix here then shows how actors are connected to relationships. By examining the rows, we can characterize how much, and in what ways actors are embedded in relations. One actor may have few entries -- a near isolate; another may have many negative and few positive entries -- a "taker" rather than a "giver." By examining the columns, we get a description of which actors are connected, in which way, by each of the relations in the graph.

Focusing on the relations, instead of the actors

Turning an actor-by-actor adjacency matrix into an actor-by-relation incidence graph takes us part of the way toward focusing on relations rather than actors. We can go further.

Transform> Linegraph converts an actor-by-actor matrix (like figure 6.4) into a full relation-by-relation matrix. Figure 6.12 shows the results of applying it to the example data.

Figure 6.12. Linegraph matrix

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

We again have a square matrix. This time, though, it describes which relations in the graph are "adjacent to" which other relations. Two relations are adjacent if they share an actor. For example, relation "1" (the tie between actors 1 and 2, or A and B) is adjacent to the relation "3" (the tie between actors 2 and 3, or B and C). Note that the "adjacency" here is directional -- relation 1 is a source of relation 3. We could also apply this approach to symmetric or simple graphs to describe which relations are simply adjacent in a un-directed way.

A quick glance at the linegraph matrix is suggestive. It is very sparse in this example -- most of the relations are not sources of other relations. The maximum degree of each entry is 1 -- no relation is the source of multiple relations. While there may be a key or central actor (A), it's not so clear that there is a single central relation.

To be entirely honest, most social network analysts do (much of the time) think about actors connected to actors by relations, rather than relations connecting actors, or relations connecting relations. But changing our point of view to put the relations first, and the actors second is, in many ways, a more distinctively "sociological" way of looking at networks. Transforming actor-by-actor data into relation-by-relation data can yield some interesting insights about social structures.

[table of contents](#)

Conclusion

In this chapter we've covered a number of rather diverse but related topics. We've described some of the basic "nuts and bolts" tools for entering and transforming network data. The "bigger picture" is to think about network data (and any other, for that matter) as having "structure." Once you begin to see data in this way, you can begin to better imagine the creative possibilities: for example, treating actor-by-attribute data as actor-by-actor, or treating it as attribute-by-attribute. Different research problems may call for quite different ways of looking at, and transforming, the same data structures. We've hardly covered every possibility here, but we have looked at some of the most frequently used tricks.

[top of this page](#)
[table of contents of the book](#)

Introduction to social network methods

7. Connection and distance

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

Contents of chapter 7: Connection and distance

- [Networks and actors](#)
 - [An example: Knoke's information exchange](#)
 - [Connection](#)
 - [Basic demographics](#)
 - [Density](#)
 - [Reachability](#)
 - [Connectivity](#)
 - [Distance](#)
 - [Walks etc.](#)
 - [Geodesic distance, eccentricity, and diameter](#)
 - [Flow](#)
 - [Summary](#)
 - [Study Questions](#)
-

Networks and actors

The social network perspective emphasizes multiple levels of analysis. Differences among actors are traced to the constraints and opportunities that arise from how they are embedded in networks; the structure and behavior of networks grounded in, and enacted by local interactions among actors. As we examine some of the basic concepts and definitions of network analysis in this and the next several chapters, this duality of individual and structure will be highlighted again and again.

In this chapter we will examine some of the most obvious and least complex ideas of formal network analysis methods. Despite the simplicity of the ideas and definitions, there are good theoretical reasons (and some empirical evidence) to believe that these basic properties of social networks have very important consequences. For both individuals and for structures, one main question is connections. Typically, some actors have lots of connections, others have fewer. Some networks are well-connected or "cohesive," others are not. The extent to which individuals are connected to others, and the extent to which the network as a whole is integrated are two sides of the same coin.

Differences among individuals in how connected they are can be extremely consequential for understanding their attributes and behavior. More connections often mean that individuals are exposed to more, and more diverse, information. Highly connected individuals may be more influential, and may be more influenced by others. Differences among whole populations in how connected they are can be quite consequential as well. Disease and rumors spread more quickly where there are high rates of connection. But, so does useful information. More connected populations may be better able to mobilize their resources, and may be better able to bring multiple and diverse perspectives to bear to solve problems. In between the individual and the whole population, there is another level of analysis -- that of "composition." Some populations may be composed of individuals who are all pretty much alike in the extent to which they are connected. Other

populations may display sharp differences, with a small elite of central and highly connected persons, and larger masses of persons with fewer connections. Differences in connections can tell us a good bit about the stratification order of social groups. A great deal of recent work by Duncan Watts, Doug White and many others outside of the social sciences is focusing on the consequences of variation in the degree of connection of actors.

Because most individuals are not usually connected directly to most other individuals in a population, it can be quite important to go beyond simply examining the immediate connections of actors, and the overall density of direct connections in populations. The second major (but closely related) set of approaches that we will examine in this chapter have to do with the idea of the distance between actors (or, conversely how close they are to one another). Some actors may be able to reach most other members of the population with little effort: they tell their friends, who tell their friends, and "everyone" knows. Other actors may have difficulty being heard. They may tell people, but the people they tell are not well connected, and the message doesn't go far. Thinking about it the other way around, if all of my friends have one another as friends, my network is fairly limited -- even though I may have quite a few friends. But, if my friends have many non-overlapping connections, the range of my connection is expanded. If individuals differ in their closeness to other actors, then the possibility of stratification along this dimension arises. Indeed, one major difference among "social classes" is not so much in the number of connections that actors have, but in whether these connections overlap and "constrain" or extent outward and provide "opportunity." Populations as a whole, then, can also differ in how close actors are to other actors, on the average. Such differences may help us to understand diffusion, homogeneity, solidarity, and other differences in macro properties of social groups.

Social network methods have a vocabulary for describing connectedness and distance that might, at first, seem rather formal and abstract. This is not surprising, as many of the ideas are taken directly from the mathematical theory of graphs. But it is worth the effort to deal with the jargon. The precision and rigor of the definitions allow us to communicate more clearly about important properties of social structures -- and often lead to insights that we would not have had if we used less formal approaches.

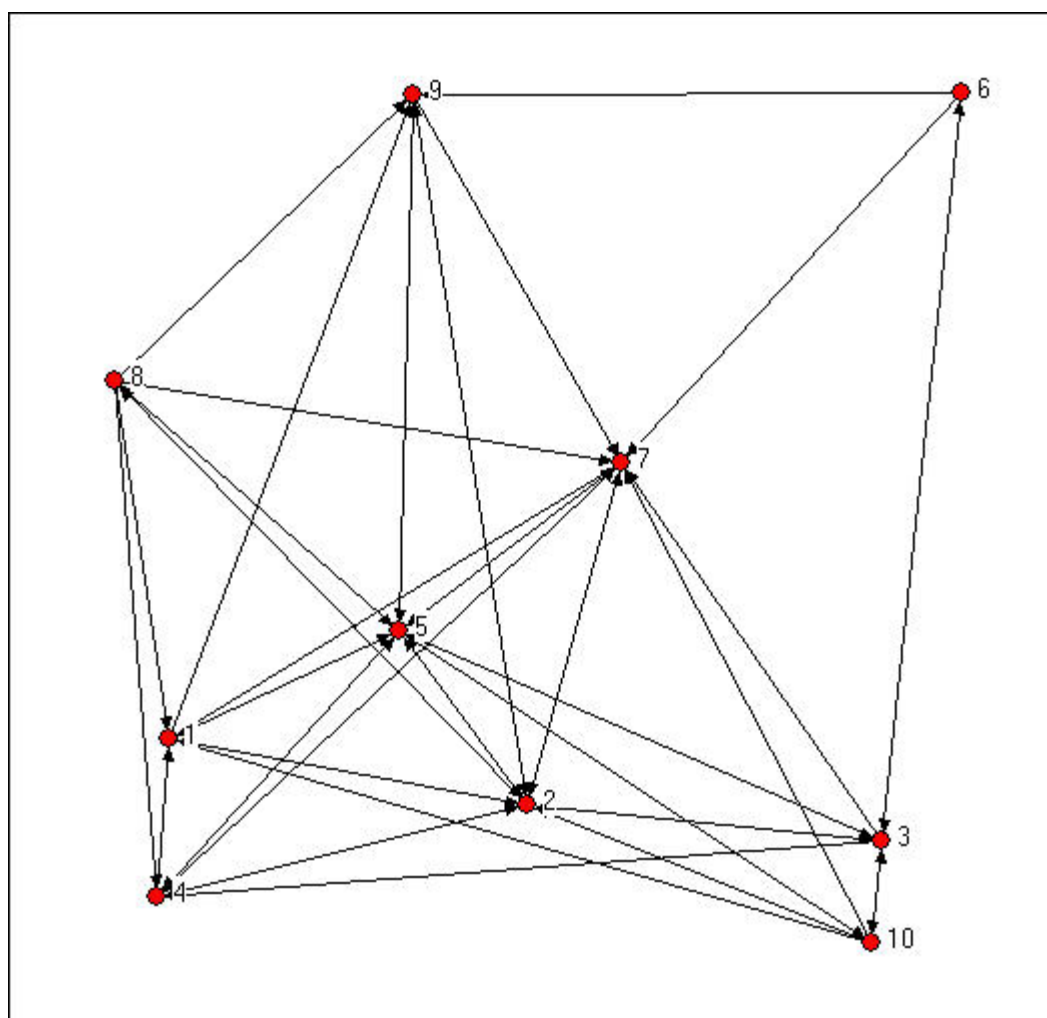
[table of contents](#)

An example: Knoke's information exchange

The basic properties of networks are easier to learn and understand by example. Studying an example also shows sociologically meaningful applications of the formalisms. In this chapter, we will look at a single directed binary network that describes the flow of information among 10 formal organizations concerned with social welfare issues in one mid-western U.S. city (Knoke and Burke). Of course, network data come in many forms (undirected, multiple ties, valued ties, etc.) and one example can't capture all of the possibilities. Still, it can be rather surprising how much information can be "squeezed out" of a single binary matrix by using basic graph concepts.

For small networks, it is often useful to examine graphs. Figure 7.1 shows the di-graph (directed graph) for the Knoke information exchange data:

Figure 7.1 Knoke information exchange directed graph



Your trained eye should immediately perceive a number of things in looking at the graph. There are a limited number of actors here (ten, actually), and all of them are "connected." But, clearly not every possible connection is present, and there are "structural holes" (or at least "thin spots" in the fabric). There appear to be some differences among the actors in how connected they are (compare actor number 7, a newspaper, to actor number 6, a welfare rights advocacy organization). If you look closely, you can see that some actor's connections are likely to be reciprocated (that is, if A shares information with B, B also shares information with A); some other actors (e.g. 6 and 10, are more likely to be senders than receivers of information). As a result of the variation in how connected individuals are, and whether the ties are reciprocated, some actors may be at quite some "distance" from other actors. There appear to be groups of actors who differ in this regard (2, 5, and 7 seem to be in the center of the action, 6, 9, and 10 seem to be more peripheral).

A careful look at the graph can be very useful in getting an intuitive grasp of the important features of a social network. With larger populations or more connections, however, graphs may not be much help. Looking at a graph can give a good intuitive sense of what is going on, but our descriptions of what we see are rather imprecise (the previous paragraph is an example of this). To get more precise, and to use computers to apply algorithms to calculate mathematical measures of graph properties, it is necessary to work with the adjacency matrix instead of the graph. The Knoke data graphed above are shown as an asymmetric adjacency matrix in figure 7.2.

Figure 7.2 Knoke information exchange adjacency matrix

Matrix #1: KNOKI										
	1	2	3	4	5	6	7	8	9	10
	C	C	E	I	M	W	N	U	W	W
1	0	1	0	0	1	0	1	0	1	0
2	1	0	1	1	1	0	1	1	1	0
3	0	1	0	1	1	1	1	0	0	1
4	1	1	0	0	1	0	1	0	0	0
5	1	1	1	1	0	0	1	1	1	1
6	0	0	1	0	0	0	1	0	1	0
7	0	1	0	1	1	0	0	0	0	0
8	1	1	0	1	1	0	1	0	1	0
9	0	1	0	0	1	0	1	0	0	0
10	1	1	1	0	1	0	1	0	0	0

Using [Data>Display](#), we can look at the network in matrix form. There are ten rows and columns, the data are binary, and the matrix is asymmetric. As we mentioned in the chapter on using matrices to represent networks, the row is treated as the source of information and the column as the receiver. By doing some very simple operations on this matrix it is possible to develop systematic and useful index numbers, or measures, of some of the network properties that our eye discerns in the graph.

[table of contents](#)

Connection

Since networks are defined by their actors and the connections among them, it is useful to begin our description of networks by examining these very simple properties. Focusing first on the network as a whole, one might be interested in the number of actors, the number of connections that are possible, and the number of connections that are actually present. Differences in the size of networks, and how connected the actors are tell us two things about human populations that are critical. Small groups differ from large groups in many important ways -- indeed, population size is one of the most critical variables in all sociological analyses. Differences in how connected the actors in a population are may be a key indicator of the "cohesion," "solidarity," "moral density," and "complexity" of the social organization of a population.

Individuals, as well as whole networks, differ in these basic demographic features. Individual actors may have many or few ties. Individuals may be "sources" of ties, "sinks" (actors that receive ties, but don't send them), or both. These kinds of very basic differences among actors immediate connections may be critical in explaining how they view the world, and how the world views them. The number and kinds of ties that actors have are a basis for similarity or dissimilarity to other actors -- and hence to possible differentiation and stratification. The number and kinds of ties that actors have are keys to determining how much their embeddedness in the network constrains their behavior, and the range of opportunities, influence, and power that they have.

[table of contents](#)

Basic demographics

Network size. The size of a network is often very important. Imagine a group of 12 students in a seminar. It would not be difficult for each of the students to know each of the others fairly well, and build up exchange relationships (e.g. sharing reading notes). Now imagine a large lecture class of 300 students. It would be extremely difficult for any student to know all of the others, and it would be virtually impossible for there to be a single network for exchanging reading notes. Size is critical for the structure of social relations because of the limited resources and capacities that each actor has for building and maintaining ties. Our example network has ten actors. Usually the size of a network is indexed simply by counting the number of nodes.

In any network there are $(k * k-1)$ unique ordered pairs of actors (that is AB is different from BA, and leaving aside self-ties), where k is the number of actors. You may wish to verify this for yourself with some small networks. So, in our network of 10 actors, with directed data, there are 90 logically possible relationships. If we had undirected, or symmetric ties, the number would be 45, since the relationship AB would be the same as BA. The number of logically possible relationships then grows exponentially as the number of actors increases linearly. It follows from this that the range of logically possible social structures increases (or, by one definition, "complexity" increases) exponentially with size.

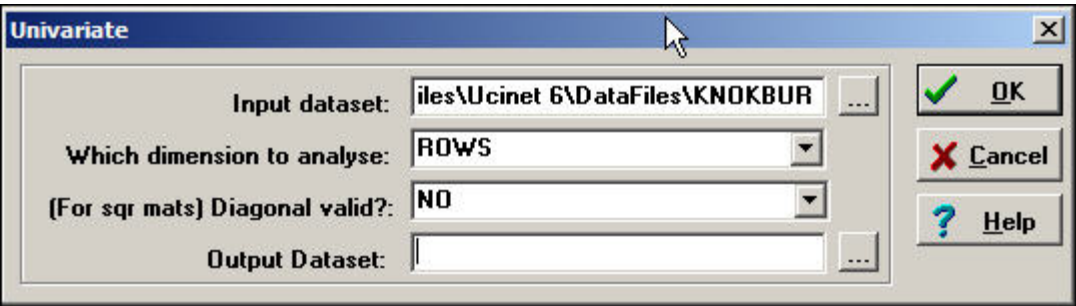
Actor degree. The number of actors places an upper limit on the number of connections that each individual can have $(k-1)$. For networks of any size, though, few -- if any -- actors approach this limit. It can be quite useful to examine the distribution of actor degree. The distribution of how connected individual actors are can tell us a good bit about the social structure.

Since the data in our example are asymmetric (that is directed ties), we can distinguish between ties being sent and ties being received. Looking at the density for each row and for each column can tell us a good bit about the way in which actors are embedded in the overall density.

Tools>Univariate Stats provides quick summaries of the distribution of actor's ties.

Let's first examine these statistics for the rows, or out-degree of actors.

Figure 7.3. Dialog for Tools>Univariate Stats



Produces this result:

Figure 7.4. Out-degree statistics for Knoke information exchange

Descriptive Statistics											
	1	2	3	4	5	6	7	8	9	10	
	Mean	Std D	Sum	Varia	SSQ	MCSSQ	Euc N	Minim	Maxim	N of	
1	0.444	0.497	4.000	0.247	4.000	2.222	2.000	0.000	1.000	9.000	
2	0.778	0.416	7.000	0.173	7.000	1.556	2.646	0.000	1.000	9.000	
3	0.667	0.471	6.000	0.222	6.000	2.000	2.449	0.000	1.000	9.000	
4	0.444	0.497	4.000	0.247	4.000	2.222	2.000	0.000	1.000	9.000	
5	0.889	0.314	8.000	0.099	8.000	0.889	2.828	0.000	1.000	9.000	
6	0.333	0.471	3.000	0.222	3.000	2.000	1.732	0.000	1.000	9.000	
7	0.333	0.471	3.000	0.222	3.000	2.000	1.732	0.000	1.000	9.000	
8	0.667	0.471	6.000	0.222	6.000	2.000	2.449	0.000	1.000	9.000	
9	0.333	0.471	3.000	0.222	3.000	2.000	1.732	0.000	1.000	9.000	
10	0.556	0.497	5.000	0.247	5.000	2.222	2.236	0.000	1.000	9.000	

Statistics on the rows tell us about the role that each actor plays as a "source" of ties (in a directed graph). The sum of the connections from the actor to others (e.g. actor #1 sends information to four others) is called the *out-degree* of the point (for symmetric data, of course, each node simply has *degree*, as we cannot distinguish *in-degree* from *out-degree*). The degree of points is important because it tells us how many

connections an actor has. With out-degree, it is usually a measure of how influential the actor may be.

We can see that actor #5 sends ties to all but one of the remaining actors; actors #6, #7 and #9 send information to only three other actors. Actors #2, #3, #5, and #8 are similar in being sources of information for large portions of the network; actors #1, #6, #7, and #9 as being similar as not being sources of information. We might predict that the first set of organizations will have specialized divisions for public relations, the latter set might not. Actors in the first set have a higher potential to be influential; actors in the latter set have lower potential to be influential; actors in "the middle" will be influential if they are connected to the "right" other actors, otherwise, they might have very little influence. So, there is variation in the roles that these organizations play as sources of information. We can norm this information (so we can compare it to other networks of different sizes, by expressing the out-degree of each point as a proportion of the number of elements in the row. That is, calculating the mean. Actor #10, for example, sends ties to 56% of the remaining actors. This is a figure we can compare across networks of different sizes.

Another way of thinking about each actor as a source of information is to look at the row-wise variance or standard deviation. We note that actors with very few out-ties, or very many out-ties have less variability than those with medium levels of ties. This tells us something: those actors with ties to almost everyone else, or with ties to almost no-one else are more "predictable" in their behavior toward any given other actor than those with intermediate numbers of ties. In a sense, actors with many ties (at the center of a network) and actors at the periphery of a network (few ties) have patterns of behavior that are more constrained and predictable. Actors with only some ties can vary more in their behavior, depending on to whom they are connected.

If we were examining a valued relation instead of a binary one, the meaning of the "sum," "mean," and "standard deviation" of actor's out-degree would differ. If the values of the relations are all positive and reflect the strength or probability of a tie between nodes, these statistics would have the easy interpretations as the sum of the strengths, the average strength, and variation in strength.

It's useful to examine the statistics for in-degree, as well (look at the data column-wise). Now, we are looking at the actors as "sinks" or receivers of information. The sum of each column in the adjacency matrix is the *in-degree of the point*. That is, how many other actors send information or ties to the one we are focusing on. Actors that receive information from many sources may be prestigious (other actors want to be known by the actor, so they send information). Actors that receive information from many sources may also be more powerful -- to the extent that "knowledge is power." But, actors that receive a lot of information could also suffer from "information overload" or "noise and interference" due to contradictory messages from different sources.

Here are the results of [Tools>Univariate Stats](#) when we select "column" instead of "row."

Figure7.5. In-degree statistics for Knoke information exchange

Descriptive Statistics											
		1	2	3	4	5	6	7	8	9	10
		COUN	COMM	EDUC	INDU	MAYR	WRO	NEWS	UWAY	WELF	WEST
1	Mean	0.556	0.889	0.444	0.556	0.889	0.111	1.000	0.222	0.556	0.222
2	Std Dev	0.497	0.314	0.497	0.497	0.314	0.314	0.000	0.416	0.497	0.416
3	Sum	5.000	8.000	4.000	5.000	8.000	1.000	9.000	2.000	5.000	2.000
4	Variance	0.247	0.099	0.247	0.247	0.099	0.099	0.000	0.173	0.247	0.173
5	SSQ	5.000	8.000	4.000	5.000	8.000	1.000	9.000	2.000	5.000	2.000
6	MCSSQ	2.222	0.889	2.222	2.222	0.889	0.889	0.000	1.556	2.222	1.556
7	Euc Norm	2.236	2.828	2.000	2.236	2.828	1.000	3.000	1.414	2.236	1.414
8	Minimum	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000
9	Maximum	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
10	N of Obs	9.000	9.000	9.000	9.000	9.000	9.000	9.000	9.000	9.000	9.000

Looking at the means, we see that there is a lot of variation in information receiving -- more than for information sending. We see that actors #2, #5, and #7 are very high. #2 and #5 are also high in sending information -- so perhaps they act as "communicators" and "facilitators" in the system. Actor #7 receives a lot of information, but does not send a lot. Actor #7, as it turns out is an "information sink" -- it collects facts, but it does not create them (at least we hope so, since actor #7 is a newspaper). Actors #6, #8, and #10 appear to be "out of the loop" -- that is, they do not receive information from many sources directly. Actor #6 also does not send much information -- so #6 appears to be something of an "isolate." Numbers #8 and #10 send relatively more information than they receive. One might suggest that they are "outsiders" who are attempting to be influential, but may be "clueless."

We can learn a great deal about a network overall, and about the structural constraints on individual actors, and even start forming some hypotheses about social roles and behavioral tendencies, just by looking at the simple adjacencies and calculating a few very basic statistics. Before discussing the slightly more complex idea of distance, there are a couple other aspects of "connectedness" that are sometimes of interest.

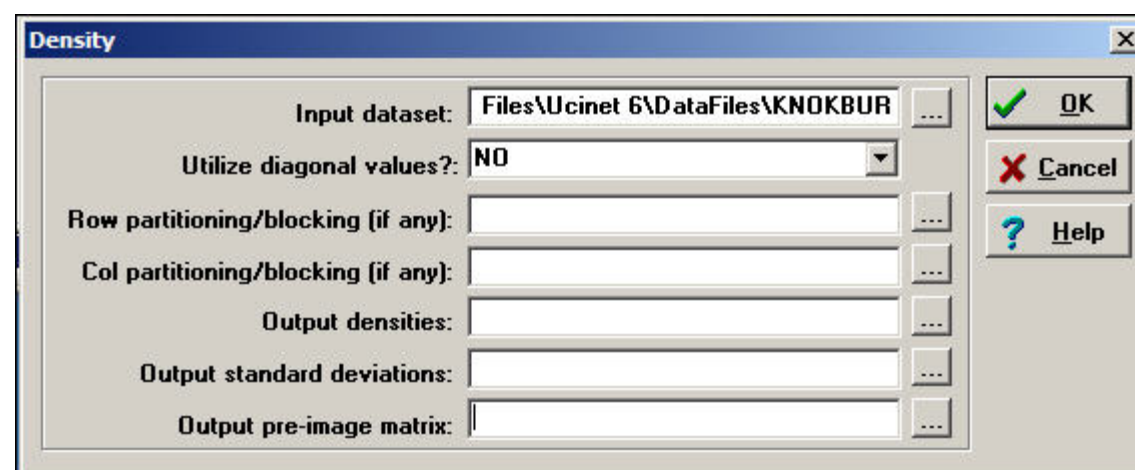
[table of contents](#)

Density

The density of a binary network is simply the proportion of all possible ties that are actually present. For a valued network, density is defined as the sum of the ties divided by the number of possible ties (i.e. the ratio of all tie strength that is actually present to the number of possible ties). The density of a network may give us insights into such phenomena as the speed at which information diffuses among the nodes, and the extent to which actors have high levels of social capital and/or social constraint.

[Network>Cohesion>Density](#) is a quite powerful tool for calculating densities. its dialog is shown in figure 7.6.

Figure 7.6 Dialog for Network>Cohesion>Density



To obtain densities for a matrix (as we are doing in this example), we simply need a dataset. Usually self-ties are ignored in computing density (but there are circumstances where you might want to include them). The [Network>Cohesion>Density](#) algorithm also can be used to calculate the densities within partitions or blocks by specifying the file name of an attribute data set that contains the node name and partition number. That is, the density tool can be used to calculate within and between block densities for data that are grouped. One might, for example, partition the Knoke data into "public" and "private" organizations, and examine the density of information exchange within and between types.

For our current purposes, we won't block or partition the data. Here's the result of the dialog above.

Figure 7.7 Density of Knoke information network

```

Relation: KNOKI
Density (matrix average) = 0.5444
Standard deviation = 0.4980

Relation: KNOKM
Density (matrix average) = 0.2444
Standard deviation = 0.4298
    
```

Since the Knoke data set contains two matrices, separate reports for each relation (KNOKI and KNOKM) are produced.

The density of the information exchange relation matrix is .5444. That is 54% of all the possible ties are present. The standard deviation of the entries in the matrix is also given. For binary data, the standard deviation is largely irrelevant -- as the standard deviation of a binary variable is a function of its mean.

Reachability

An actor is "reachable" by another if there exists any set of connections by which we can trace from the source to the target actor, regardless of how many others fall between them. If the data are asymmetric or directed, it is possible that actor A can reach actor B, but that actor B cannot reach actor A. With symmetric or undirected data, of course, each pair of actors either are or are not reachable to one another. If some actors in a network cannot reach others, there is the potential of a division of the network. Or, it may indicate that the population we are studying is really composed of more than one sub-populations.

In the Knoke information exchange data set, it turns out that all actors are reachable by all others. This is something that you can verify by eye. See if you can find any pair of actors in the diagram such that you cannot trace from the first to the second along arrows all headed in the same direction (don't waste a lot of time on this, there is no such pair!). For the Knoke "M" relation, it turns out that not all actors can "reach" all other actors. Here's the output of [Network>Cohesion>Reachability](#) from UCINET.

Figure 7.8 Reachability of Knoke "I" and "M" relations

Matrix #1

	1	2	3	4	5	6	7	8	9	0
	C	C	E	I	M	W	N	U	W	W
1	0	1	1	1	1	1	1	1	1	1
2	1	0	1	1	1	1	1	1	1	1
3	1	1	0	1	1	1	1	1	1	1
4	1	1	1	0	1	1	1	1	1	1
5	1	1	1	1	0	1	1	1	1	1
6	1	1	1	1	1	0	1	1	1	1
7	1	1	1	1	1	1	0	1	1	1
8	1	1	1	1	1	1	1	0	1	1
9	1	1	1	1	1	1	1	1	0	1
10	1	1	1	1	1	1	1	1	1	0

Matrix #2

	1	2	3	4	5	6	7	8	9	0
	C	C	E	I	M	W	N	U	W	W
1	0	1	1	0	1	0	0	1	1	1
2	0	0	1	0	0	0	0	1	1	1
3	0	0	0	0	0	0	0	1	1	1
4	0	1	1	0	0	0	1	1	1	1
5	0	1	1	0	0	0	0	1	1	1
6	0	0	0	0	0	0	0	0	0	0
7	0	1	1	0	0	0	0	1	1	1
8	0	0	1	0	0	0	0	0	1	1
9	0	0	1	0	0	0	0	1	0	1
10	0	0	0	0	0	0	0	0	0	0

So, there exists a directed "path" from each organization to each other actor for the flow of information, but not for the flow of money. Sometimes "what goes around comes around," and sometimes it doesn't!

Connectivity

Adjacency tells us whether there is a direct connection from one actor to another (or between two actors for un-directed data). Reachability tells us whether two actors are connected or not by way of either a direct or an indirect pathways of any length.

Network>Cohesion>Point Connectivity calculates the number of nodes that would have to be removed in order for one actor to no longer be able to reach another. If there are many different pathways that connect two actors, they have high "connectivity" in the sense that there are multiple ways for a signal to reach from one to the other. Figure 7.9 shows the point connectivity for the flow information among the 10 Knoke organizations.

Figure 7.9. Point connectivity of Knoke information exchange

	1	2	3	4	5	6	7	8	9	10
	C	C	E	I	M	W	N	U	W	W
1	5	5	3	4	5	1	6	4	4	3
2	5	8	3	5	8	1	6	5	3	4
3	3	3	4	4	3	1	4	3	3	3
4	5	5	3	5	5	1	5	4	3	4
5	5	8	3	5	8	1	6	5	3	5
6	1	1	1	1	1	1	2	1	2	1
7	5	6	3	5	6	1	6	4	2	3
8	5	5	3	5	5	1	5	5	4	4
9	3	3	3	3	3	1	3	3	3	3
10	4	5	3	4	5	1	4	4	3	5

The result again demonstrates the tenuousness of organization 6's connection as both a source (row) or receiver (column) of information. To get its message to most other actors, organization 6 has alternative; should a single organization refuse to pass along information, organization 6 would receive none at all! Point connectivity can be a useful measure to get at notions of dependency and vulnerability.

[table of contents](#)

Distance

The properties of the network that we have examined so far primarily deal with adjacencies -- the direct connections from one actor to the next. But the way that people are embedded in networks is more complex than this. Two persons, call them A and B, might each have five friends. But suppose that none of person A's friends have any friends except A. Person B's five friends, in contrast, each have five friends. The information available to B, and B's potential for influence is far greater than A's. That is, sometimes being a "friend of a friend" may be quite consequential.

To capture this aspect of how individuals are embedded in networks, one main approach is to examine the distance that an actor is from others. If two actors are adjacent, the distance between them is one (that is, it takes one step for a signal to go from the source to the receiver). If A tells B, and B tells C (and A does not tell C), then actors A and C are at a distance of two. How many actors are at various distances from each actor can be important for understanding the differences among actors in the constraints and opportunities they have as a result of their position. Sometimes we are also interested in how many ways there are to connect between two actors, at a given distance. That is, can actor A reach actor B in more than one way? Sometimes multiple connections may indicate a stronger connection between two actors than a single connection.

The distances among actors in a network may be an important macro-characteristic of the network as a whole. Where distances are great, it may take a long time for information to diffuse across a population. It may also be that some actors are quite unaware of, and influenced by others -- even if they are technically reachable, the costs may be too high to conduct exchanges. The variability across the actors in the distances that they have from other actors may be a basis for differentiation and even stratification. Those actors who are closer to more others may be able to exert more power than those who are more distant. We will have a good deal more to say about this aspect of variability in actor distances in the next chapter.

For the moment, we need to learn a bit of jargon that is used to describe the distances between actors: *walks*, *paths*, *semi-paths*, etc. Using these basic definitions, we can then develop some more powerful ways of describing various aspects of the distances among actors in a network.

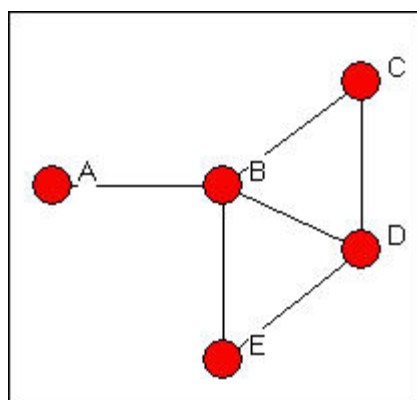
[table of contents](#)

Walks etc.

To describe the distances between actors in a network with precision, we need some terminology. And, as it turns out, whether we are talking about a simple graph or a directed graph makes a good bit of difference. If A and B are adjacent in a simple graph, they have a distance of one. In a directed graph, however, A can be adjacent to B while B is not adjacent to A -- the distance from A to B is one, but there is no distance from B to A. Because of this difference, we need slightly different terms to describe distances between actors in graphs and digraphs.

Simple graphs: The most general form of connection between two actors in a graph is called a *walk*. A walk is a sequence of actors and relations that begins and ends with actors. A *closed walk* is one where the beginning and end point of the walk are the same actor. Walks are unrestricted. A walk can involve the same actor or the same relation multiple times. A *cycle* is a specially restricted walk that is often used in algorithms examining the neighborhoods (the points adjacent) of actors. A cycle is a closed walk of 3 or more actors, all of whom are distinct, except for the origin/destination actor. The length of a walk is simply the number of relations contained in it. For example, consider this graph in figure 7.10.

Figure 7.10. Walks in a simple graph



There are many walks in a graph (actually, an infinite number if we are willing to include walks of any length - though, usually, we restrict our attention to fairly small lengths). To illustrate just a few, begin at actor A and go to actor C. There is one walk of length 2 (A,B,C). There is one walk of length three (A,B,D,C). There are several walks of length four (A,B,E,D,C; A,B,D,B,C; A,B,E,B,C). Because these are unrestricted, the same actors and relations can be used more than once in a given walk. There are no cycles beginning and ending with A. There are some beginning and ending with actor B (B,D,C,B; B,E,D,B; B,C,D,E,B).

It is usually more useful to restrict our notion of what constitutes a connection somewhat. One possibility is to restrict the count only walks that do not re-use relations. A *trail* between two actors is any walk that includes a given relation no more than once (the same other actors, however, can be part of a trail multiple times). The length of a trail is the number of relations in it. All trails are walks, but not all walks are trails. If the trail begins and ends with the same actor, it is called a *closed trail*. In our example above, there are a number of trails from A to C. Excluded are tracings like A,B,D,B,C (which is a walk, but is not a trail because the relation BD is used more than once).

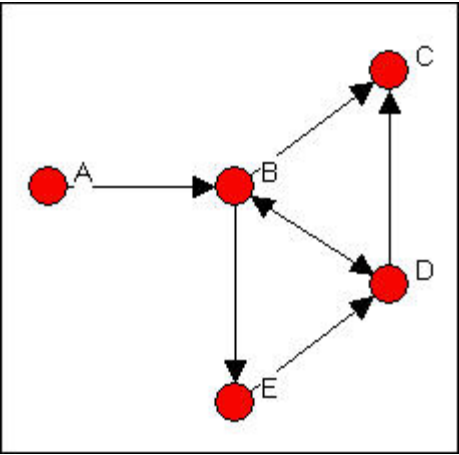
Perhaps the most useful definition of a connection between two actors (or between an actor and themselves) is a *path*. A path is a walk in which each other actor and each other relation in the graph may be used at most one time. The single exception to this is a *closed path*, which begins and ends with the same actor. All paths are trails and walks, but all walks and all trails are not paths. In our example, there are a limited number of paths connecting A and C: A,B,C; A,B,D,C; A,B,E,D,C.

Directed graphs: Walks, trails, and paths can also be defined for directed graphs. But there are two flavors

of each, depending on whether we want to take direction into account or not . *Semi-walks, semi-trails, and semi-paths* are the same as for undirected data. In defining these distances, the directionality of connections is simply ignored (that is, arcs - or directed ties are treated as though they were edges - undirected ties). As always, the length of these distances is the number of relations in the walk, trail, or path.

If we do want to pay attention to the directionality of the connections we can define *walks, trails, and paths* in the same way as before, but with the restriction that we may not "change direction" as we move across relations from actor to actor. Consider the directed graph in figure 7.11

Figure 7.11. Walks in a directed graph



In this directed graph, there are a number of walks from A to C. However, there are no walks from C (or anywhere else) to A. Some of these walks from A to C are also trails (e.g. A,B,E,D,B,C). There are, however, only three paths from A to C. One path is length 2 (A,B,C); one is length three (A,B,D,C); one is length four (A,B,E,D,C).

The various kinds of connections (walks, trails, paths) provide us with a number of different ways of thinking about the distances between actors. The main reason that social network analysts are concerned with these distances is that they provide a way of thinking about the strength of ties or relations. Actors that are connected at short lengths or distances may have stronger connections; actors that are connected many times (for example, having many, rather than a single path) may have stronger ties. Their connection may also be less subject to disruption, and hence more stable and reliable.

The numbers of walks of a given length between all pairs of actors can be found by raising the matrix to that power. A convenient method for accomplishing this is to use *Tools>Matrix Algebra*, and to specify an expression like *out=prod(X1,X1)*. This produces the square of the matrix X1, and stores it as the data set "out." A more detailed discussion of this idea can be found in the earlier chapter on representing networks as matrices. This matrix could then be added to X1 to show the number of walks between any two actors of length two or less.

Let's look briefly at the distances between pairs of actors in the Knoke data on directed information flows. Counts of the numbers of paths of various lengths are shown in figure 7.12.

Figure 7.12. Numbers of walks in Knoke information network

# of walks of length 1										
	1	2	3	4	5	6	7	8	9	0
1	0	1	0	0	1	0	1	0	1	0
2	1	0	1	1	1	0	1	1	1	0
3	0	1	0	1	1	1	1	0	0	1

4	1	1	0	0	1	0	1	0	0	0
5	1	1	1	1	0	0	1	1	1	1
6	0	0	1	0	0	0	1	0	1	0
7	0	1	0	1	1	0	0	0	0	0
8	1	1	0	1	1	0	1	0	1	0
9	0	1	0	0	1	0	1	0	0	0
10	1	1	1	0	1	0	1	0	0	0

# of walks of length 2										
	1	2	3	4	5	6	7	8	9	10
	-	-	-	-	-	-	-	-	-	-
1	2	3	2	3	3	0	3	2	2	1
2	3	7	1	4	6	1	6	1	3	2
3	4	4	4	3	4	0	5	2	3	1
4	2	3	2	3	3	0	3	2	3	1
5	4	7	2	4	8	1	7	1	3	1
6	0	3	0	2	3	1	2	0	0	1
7	3	2	2	2	2	0	3	2	2	1
8	3	5	2	3	5	0	5	2	3	1
9	2	2	2	3	2	0	2	2	2	1
10	2	4	2	4	4	1	4	2	3	2

# of walks of length 3										
	1	2	3	4	5	6	7	8	9	10
	-	-	-	-	-	-	-	-	-	-
1	12	18	7	13	18	2	18	6	10	5
2	20	26	16	21	27	1	28	13	18	7
3	14	26	9	19	26	4	25	8	14	8
4	12	19	7	13	19	2	19	6	10	5
5	21	30	17	25	29	2	31	15	21	10
6	9	8	8	8	8	0	10	6	7	3
7	9	17	5	11	17	2	16	4	9	4
8	16	24	11	19	24	2	24	10	15	7
9	10	16	5	10	16	2	16	4	8	4
10	16	23	11	16	23	2	24	8	13	6

Total number of walks (lengths 1, 2, 3)										
	1	2	3	4	5	6	7	8	9	10
	-	-	-	-	-	-	-	-	-	-
1	14	21	9	16	21	2	21	8	12	6
2	23	33	17	25	33	2	34	14	21	9
3	18	30	13	22	30	4	30	10	17	9
4	14	22	9	16	22	2	22	8	13	6
5	25	37	19	29	37	3	38	16	24	11
6	9	11	8	10	11	1	12	6	7	4
7	12	19	7	13	19	2	19	6	11	5
8	19	29	13	22	29	2	29	12	18	8
9	12	18	7	13	18	2	18	6	10	5
10	18	27	13	20	27	3	28	10	16	8

The inventory of the total connections among actors is primarily useful for getting a sense of how "close" each pair is, and for getting a sense of how closely coupled the entire system is. Here, we can see that using only connections of two steps (e.g. "A friend of a friend"), there is a great deal of connection in the graph overall; we also see that there are sharp differences among actors in their degree of connectedness, and who they are connected to. These differences can be used to understand how information moves in the network, which actors are likely to be influential on one another, and a number of other important properties.

[table of contents](#)

Geodesic distance, eccentricity, and diameter

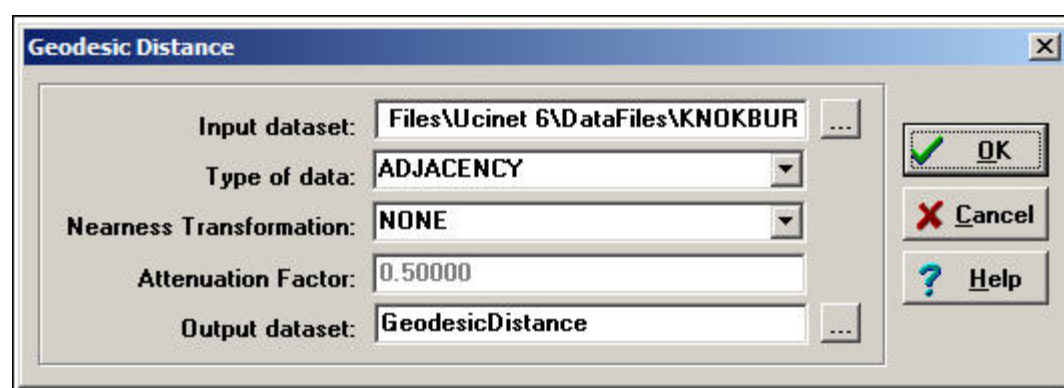
One particular definition of the distance between actors in a network is used by most algorithms to define more complex properties of individual's positions and the structure of the network as a whole. This quantity is the *geodesic distance*. For both directed and undirected data, the geodesic distance is the number of relations in the shortest possible walk from one actor to another (or, from an actor to themselves, if we care, which we usually do not).

The geodesic distance is widely used in network analysis. There may be many connections between two

actors in a network. If we consider how the relation between two actors may provide each with opportunity and constraint, it may well be the case that not all of these ties matter. For example, suppose that I am trying to send a message to Sue. Since I know her e-mail address, I can send it directly (a path of length 1). I also know Donna, and I know that Donna has Sue's email address. I could send my message for Sue to Donna, and ask her to forward it. This would be a path of length two. Confronted with this choice, I am likely to choose the geodesic path (i.e. directly to Sue) because it is less trouble and faster, and because it does not depend on Donna. That is, the geodesic path (or paths, as there can be more than one) is often the "optimal" or most "efficient" connection between two actors. Many algorithms in network analysis assume that actors will use the geodesic path when alternatives are available.

Using UCINET, we can easily locate the lengths of the geodesic paths in our directed data on information exchanges. Here is the dialog box for [Network>Cohesion>Distance](#).

Figure 7.13. Network>Cohesion>Distance dialog



The Knoke information exchange data are binary (organization A sends information to organization B, or it doesn't). That is, the pattern is summarized by an adjacency matrix. For binary data, the geodesic distance between two actors is the count of the number of links in the shortest path between them.

It is also possible to define the distance between two actors where the links are valued. That is, where we have a measure of the strength of ties, the opportunity costs of ties, or the probability of a tie.

[Network>Cohesion>Distance](#) can calculate distance (and nearness) for valued data, as well (select the appropriate "type of data").

Where we have measures of the strengths of ties (e.g. the dollar volume of trade between two nations), the "distance" between two actors is defined as the strength of the weakest path between them. If A sends 6 units to B, and B sends 4 units to C, the "strength" of the path from A to C (assuming A to B to C is the shortest path) is 4.

Where we have a measure of the cost of making a connection (as in an "opportunity cost" or "transaction cost" analysis), the "distance" between two actors is defined as the sum of the costs along the shortest pathway.

Where we have a measure of the probability that a link will be used, the "distance" between two actors is defined as the product along the pathway -- as in path analysis in statistics.

The *Nearness Transformation* and *Attenuation Factor* parts of the dialog allow the rescaling of distances into near-nesses. For many analyses, we may be interesting in thinking about the connections among actors in terms of how close or similar they are, rather than how distant. There are a number of ways that this may be done.

The *multiplicative* nearness transformation divides the distance by the largest possible distance between two actors. For example, if we had 7 nodes, the maximum possible distance for adjacency data would be 6. This method gives a measure of the distance as a percentage of the theoretical maximum for a given graph.

The *additive* nearness transformation subtracts the actual distance between two actors from the number of nodes. It is similar to the multiplicative scaling, but yields a value as the nearness measure, rather than a proportion.

The *linear* nearness transformation rescales distance by reversing the scale (i.e. the closest becomes the most distant, the most distant becomes the nearest) and re-scoring to make the scale range from zero (closest pair of nodes) to one (most distant pair of nodes).

The *exponential decay* method turns distance into nearness by weighting the links in the pathway with decreasing values as they fall farther away from ego. With an *attenuation factor* of .5, for example, a path from A to B to C would result in a distance of 1.5.

The *frequency decay* method is defined as 1 minus the proportion of other actors who are as close or closer to the target as ego is. The idea (Ron Burt's) is that if there are many other actors closer to the target you are trying to reach than yourself, you are effectively "more distant."

In our example, we are using simple directed adjacencies, and the results (figure 7.14) are quite straightforward.

Figure 7.14. Geodesic distances for Knoke information exchange

Geodesic Distances										
	1	2	3	4	5	6	7	8	9	10
	C	C	E	I	M	W	N	U	W	W
1	0	1	2	2	1	3	1	2	1	2
2	1	0	1	1	1	2	1	1	1	2
3	2	1	0	1	1	1	1	2	2	1
4	1	1	2	0	1	3	1	2	2	2
5	1	1	1	1	0	2	1	1	1	1
6	3	2	1	2	2	0	1	3	1	2
7	2	1	2	1	1	3	0	2	2	2
8	1	1	2	1	1	3	1	0	1	2
9	2	1	2	2	1	3	1	2	0	2
10	1	1	1	2	1	2	1	2	2	0

Because the network is moderately dense, the geodesic distances are generally small. This suggests that information may travel pretty quickly in this network. Also note that there is a geodesic distance for each x, y and y, x pair -- that is, the graph is fully connected, and all actors are "reachable" from all others (that is, there exists a path of some length from each actor to each other actor). When a network is not fully connected, we cannot exactly define the geodesic distances among all pairs. The standard approach in such cases is to treat the geodesic distance between unconnected actors as a length greater than that of any real distance in the data. For each actor, we could calculate the mean and standard deviation of their geodesic distances to describe their closeness to all other actors. For each actor, that actor's largest geodesic distance is called the *eccentricity* -- a measure of how far a actor is from the furthest other.

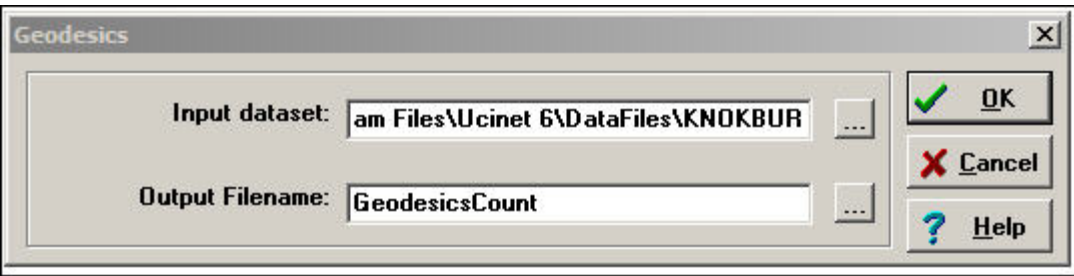
Because the current network is fully connected, a message that starts anywhere will eventually reach everyone. Although the computer has not calculated it, we might want to calculate the mean (or median) geodesic distance, and the standard deviation in geodesic distances for the matrix, and for each actor row-wise and column-wise. This would tell us how far each actor is from each other as a source of information for the other; and how far each actor is from each other actor who may be trying to influence them. It also tells

us which actors behavior (in this case, whether they've heard something or not) is most predictable and least predictable.

In looking at the whole network, we see that it is connected, and that the average geodesic distance among actors is quite small. This suggests a system in which information is likely to reach everyone, and to do so fairly quickly. To get another notion of the size of a network, we might think about its diameter. The *diameter of a network* is the largest geodesic distance in the (connected) network. In the current case, no actor is more than three steps from any other -- a very "compact" network. The diameter of a network tells us how "big" it is, in one sense (that is, how many steps are necessary to get from one side of it to the other). The diameter is also a useful quantity in that it can be used to set an upper bound on the lengths of connections that we study. Many researchers limit their explorations of the connections among actors to involve connections that are no longer than the diameter of the network.

Sometimes the redundancy of connection is an important feature of a network structure. If there are many efficient paths connecting two actors, the odds are improved that a signal will get from one to the other. One index of this is a count of the number of geodesic paths between each pair of actors. Of course, if two actors are adjacent, there can only be one such path. The number of geodesic paths can be calculated with [Network>Cohesion>No. of Geodesics](#), as in figure 7.15.

Figure 7.15. Dialog for Network>Cohesion>No. of Geodesics



The results are shown in figure 7.16.

Figure 7.16. Number of geodesic paths for Knoke information exchange

# of Geodesic Paths										
	1	2	3	4	5	6	7	8	9	10
C	1	1	1	1	1	1	1	1	1	1
C	1	1	1	1	1	1	1	1	1	1
E	1	1	1	1	1	1	1	1	1	1
I	1	1	1	1	1	1	1	1	1	1
M	1	1	1	1	1	1	1	1	1	1
W	1	1	1	1	1	1	1	1	1	1
N	1	1	1	1	1	1	1	1	1	1
U	1	1	1	1	1	1	1	1	1	1
W	1	1	1	1	1	1	1	1	1	1
W	1	1	1	1	1	1	1	1	1	1

We see that most of the geodesic connections among these actors are not only short distance, but that there are very often multiple shortest paths from x to y. This suggests a couple things: information flow is not likely to break down, because there are multiple paths; and, it will be difficult for any individual to be a powerful "broker" in this structure because most actors have alternative efficient ways of connection to other actors that can by-pass any given actor.

Flow

The use of geodesic paths to examine properties of the distances between individuals and for the whole network often makes a great deal of sense. But, there may be other cases where the distance between two actors, and the connectedness of the graph as a whole is best thought of as involving all connections -- not just the most efficient ones. If I start a rumor, for example, it will pass through a network by all pathways -- not just the most efficient ones. How much credence another person gives my rumor may depend on how many times they hear it from different sources -- and not how soon they hear it. For uses of distance like this, we need to take into account all of the connections among actors.

Several approaches have been developed for counting the amount of connection between pairs of actors that take into account all connections between them. These measures have been used for a number of different purposes, and these differences are reflected in the algorithms used to calculate them. We will examine three such ideas.

Network>Cohesion>Maximum Flow. One notion of how totally connected two actors are (called maximum flow by UCINET) asks how many different actors in the neighborhood of a source lead to pathways to a target. If I need to get a message to you, and there is only one other person to whom I can send this for retransmission, my connection is weak - even if the person I send it to may have many ways of reaching you. If, on the other hand, there are four people to whom I can send my message, each of whom has one or more ways of retransmitting my message to you, then my connection is stronger. The "flow" approach suggests that the strength of my tie to you is no stronger than the weakest link in the chain of connections, where weakness means a lack of alternatives. This approach to connection between actors is closely connected to the notion of between-ness that we will examine a bit later. It is also logically close to the idea that the number of pathways, not their length may be important in connecting people. For our directed information flow data, the results of UCINET's count of maximum flow are shown in figure 7.17.

Figure 7.17. Maximum flow for Knoke information network

	1	2	3	4	5	6	7	8	9	0
	C	C	E	I	M	W	N	U	W	W
1	0	4	3	4	4	1	4	2	4	2
2	5	0	3	5	7	1	7	2	5	2
3	5	6	0	5	6	1	6	2	5	2
4	4	4	3	0	4	1	4	2	4	2
5	5	8	3	5	0	1	8	2	5	2
6	3	3	3	3	3	0	3	2	3	2
7	3	3	3	3	3	1	0	2	3	2
8	5	6	3	5	6	1	6	0	5	2
9	3	3	3	3	3	1	3	2	0	2
10	5	5	3	5	5	1	5	2	5	0

You should verify for yourself that, for example, there are four intermediaries, or alternative routes in flows from actor 1 to actor 2, but five such points in the flow from actor 2 to actor 1. The higher the number of flows from one actor to another, the greater the likelihood that communication will occur, and the less "vulnerable" the connection. Note that actors 6, 7, and 9 are relatively disadvantaged. In particular, actor 6 has only one way of obtaining information from all other actors (the column vector of flows to actor 6).

[table of contents](#)

Summary

There is a great deal of information about both individuals and the population in a single adjacency matrix. In this chapter you have learned a lot of terminology for describing the connections and distances between actors, and for whole populations.

One focus in basic network analysis is on the immediate neighborhood of each actor: the dyads and triads in which they are involved. The degree of an actor, and the in-degree and out-degree (if the data are directed) tell us about the extent to which an actor may be constrained by, or constrain others. The extent to which an actor can reach others in the network may be useful in describing an actor's opportunity structure. We have also seen that it is possible to describe "types" of actors who may form groups or strata on the basis of their places in opportunity structures -- e.g. "isolates" "sources" etc.

Most of the time and effort of most social actors is spent in very local contexts -- interacting in dyads and triads. In looking at the connections of actors, we have suggested that the degree of "reciprocity" and "balance" and "transitivity" in relations can be regarded as important indicators of the stability and institutionalization (that is, the extent to which relations are taken for granted and are norm governed) of actor's positions in social networks.

The local connections of actors are important for understanding the social behavior of the whole population, as well as for understanding each individual. The size of the network, its density, whether all actors are reachable by all others (i.e. is the whole population connected, or are there multiple components?), whether ties tend to be reciprocal or transitive, and all the other properties that we examined for individual connections are meaningful in describing the whole population. Both the typical levels of characteristics (e.g. the mean degree of points), and the amount of diversity in characteristics (e.g. the variance in the degree of points) may be important in explaining macro behavior. Populations with high density respond differently to challenges from the environment than those with low density; populations with greater diversity in individual densities may be more likely to develop stable social differentiation and stratification.

In this chapter we also examined some properties of individual's embeddedness and of whole networks that look at the broader, rather than the local neighborhoods of actors. A set of specialized terminology was introduced to describe the distances between pairs of actors: walks, trails, and paths. We noted that there are some important differences between un-directed and directed data in applying these ideas of distance.

One of the most common and important approaches to indexing the distances between actors is the geodesic. The geodesic is useful for describing the minimum distance between actors. The geodesic distances between pairs of actors is the most commonly used measure of closeness. The average geodesic distance for an actor to all others, the variation in these distances, and the number of geodesic distances to other actors may all describe important similarities and differences between actors in how, and how closely they are connected to their entire population.

The geodesic distance, however, examines only a single connection between a pair of actors (or, in some cases several, if there are multiple geodesics connecting them). Sometimes the sum of all connections between actors, rather than the shortest connection may be relevant. We have examined approaches to measuring the vulnerability of the connection between actors by looking at the number of geodesic connections between pairs of actors, and the total number of pathways between pairs of actors.

We have seen that there is a great deal of information available in fairly simple examinations of an adjacency matrix. Life, of course, can get more complicated. We could have multiple layers, or multiplex data; we could have data that gave information on the strength of ties, rather than simple presence or absence. Nonetheless, the methods that we've used here will usually give you a pretty good grasp of what is going on in more complicated data.

Now that you have a pretty good grasp of the basics of connection and distance, you are ready to use these

ideas to build some concepts and methods for describing somewhat more complicated aspects of the network structures of populations. In the next two chapters, we will focus on ways of examining the local neighborhoods of actors. In chapter 8, we will look at methods for summarizing the entire graph in terms of the kinds of connections that individuals have to their neighbors. In chapter 9, we'll examine actors local neighborhoods from their own individual perspective.

[table of contents](#)

Review questions

1. Explain the differences among the "three levels of analysis" of graphs (individual, aggregate, whole).
2. How is the size of a network measured? Why is population size so important in sociological analysis?
3. You have a network of 5 actors, assuming no self-ties, what is the potential number of directed ties? what is the potential number of un-directed ties?
4. How is density measured? Why is density important in sociological analysis?
5. What is the "degree of a point?" Why might it be important, sociologically, if some actors have high degree and other actors have lower degree? What is the difference between "in-degree" and "out-degree?"
6. If actor "A" is reachable from actor "B" does that necessarily mean that actor "B" is reachable from actor "A?" Why or why not?
7. For pairs of actors with directed relations, there are four possible configurations of ties. Can you show these? Which configurations are "balanced?" For a triad with undirected relations, how many possible configurations of ties are there? which ones are balanced or transitive?
8. What are the differences among walks, trails, and paths? Why are "paths" the most commonly used approach to inter-actor distances in sociological analysis?
9. What is the "geodesic" distance between two actors? Many social network measures assume that the geodesic path is the most important path between actors -- why is this a plausible assumption?
10. I have two populations of ten actors each, one has a network diameter of 3, the other has a network diameter of 6. Can you explain this statement to someone who doesn't know social network analysis? Can you explain why this difference in diameter might be important in understanding differences between the two populations?
11. How do "weighted flow" approaches to social distance differ from "geodesic" approaches to social distance?
12. Why might it matter if two actors have more than one geodesic or other path between them?

Application questions

1. Think of the readings from the first part of the course. Which studies used the ideas of connectedness and density? Which studies used the ideas of distance? What specific approaches did they use to measure these concepts?
2. Draw the graphs of a "star" a "circle" a "line" and a "hierarchy." Describe the size, potential, and density of each graph. Examine the degrees of points in each graph -- are there differences among actors? Do these

differences tell us something about the "social roles" of the actors? Create a matrix for each graph that shows the geodesic distances between each pair of actors. Are there differences between the graphs in whether actors are connected by multiple geodesic distances?

3. Think about a small group of people that you know well (maybe your family, neighbors, a study group, etc.). Who helps whom in this group? What is the density of the ties? Are ties reciprocated? Are triads transitive?

4. Chrysler Corporation has called on you to be a consultant. Their research division is taking too long to generate new models of cars, and often the work of the "stylists" doesn't fit well with the work of the "manufacturing engineers" (the people who figure out how to actually build the car). Chrysler's research division is organized as a classical hierarchical bureaucracy with two branches (stylists, manufacturing) coordinated through group managers and a division manager. Analyze the reasons why performance is poor. Suggest some alternative ways of organizing that might improve performance, and explain why they will help.

[table of contents](#)

[table of contents of the book](#)

Introduction to social network methods

8. Embedding

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology](#), [University of California, Riverside](#)) and Mark Riddle (Department of Sociology, University of Northern Colorado). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail](#).

Contents of the chapter 8: Embedding

- [Introduction](#)
 - [Density](#)
 - [Reciprocity](#)
 - [Transitivity](#)
 - [Clustering](#)
 - [Group-external and group-internal ties](#)
 - [Krackhardt's Graph Theoretical Dimensions of Hierarchy](#)
 - [Summary](#)
-

Introduction

In the previous chapter we looked at some tools for examining the ways that individuals are connected, and the distances between them. In this chapter we will look at the same issue of connection. This time, though, our focus is the social structure, rather than the individual. That is, we will adopt a more "macro" perspective that focuses on the structures within which individual actors are embedded.

The "top down" perspective we'll follow in this chapter seeks to understand and describe whole populations by the "texture" of the relations that constrain its individual members. Imagine one society in which extended kin groups live in separate villages at considerable distances from one another. Most "texture" of the society will be one in which individuals have strong ties to relatively small numbers of others in local "clusters." Compare this to a society where a large portion of the population lives in a single large city. Here, the "texture" of social relations is quite different -- individuals may be embedded in smaller nuclear families of mating relations, but have diverse ties to neighbors, friends, co-workers, and others.

Social network analysts have developed a number of tools for conceptualizing and indexing the variations in the kinds of structures that characterize populations. In this chapter, we'll examine a few of these tools.

The smallest social structure in which an individual can be embedded is a dyad (that is, a pair of actors). For binary ties (present or absent), there are two possibilities for each pair in the population - either they have a tie, or they don't. We can characterize the whole population in terms of the prevalence of these dyadic "structures." This is what the density measure does.

If we are considering a directed relation (A might like B, but B might not like A), there are three kinds of dyads (no tie, one likes the other but not vice versa, or both like the other). The extent to which a population is characterized by "reciprocated" ties (those where each directs a tie to the other) may tell us about the degree of cohesion, trust, and social capital that is present.

The smallest social structure that has the true character of a "society" is the triad - any "triple" {A, B, C} of actors. Such a structure "embeds" dyadic relations in a structure where "other" is present along with "ego"

and "alter." The analysis of triads, and the prevalence of different types of triads in populations has been a staple of sociometry and social network analysis. In (directed) triads, we can see the emergence of tendencies toward equilibrium and consistency -- institutionalization -- of social structures (balance and transitivity). Triads are also the simplest structures in which we can see the emergence of hierarchy.

Most of the time, most people interact with a fairly small set of others, many of whom know one another. The extent of local "clustering" in populations can be quite informative about the texture of everyday life. Actors are also embedded in "categorical social units" or "sub-populations" defined either by shared attributes or shared membership. The extent to which these sub-populations are open or closed - the extent to which most individuals have most of their ties lives within the boundaries of these groups - may be a telling dimension of social structure.

There are many approaches to characterizing the extent and form of "embedding" of actors in populations. There is no one "right" way of indexing the degree of embedding in a population that will be effective for all analytic purposes. There are, however, some very interesting and often useful approaches that you may wish to explore.

[table of contents](#)

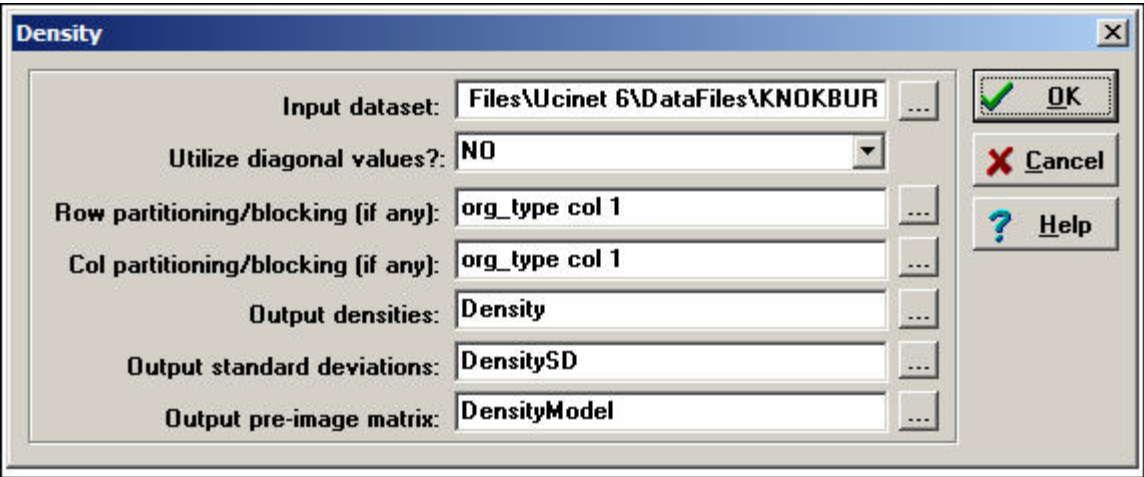
Density

If we are comparing two populations, and we note that there are many actors in one that are not connected to any other ("isolates"), and in the other population most actors are embedded in at least one dyad -- we would likely conclude that social life is very different in the two populations.

Measuring the density of a network gives us a ready index of the degree of dyadic connection in a population. For binary data, density is simply the ratio of the number of adjacencies that are present divided by the number of pairs - what proportion of all possible dyadic connections are actually present. If we have measured the ties among actors with values (strengths, closeness, probabilities, etc.) density is usually defined as the sum of the values of all ties divided by the number of possible ties. That is, with valued data, density is usually defined as the average strength of ties across all possible (not all actual) ties. Where the data are symmetric or un-directed, density is calculated relative to the number of unique pairs $((n*n-1)/2)$; where the data are directed, density is calculated across the total number of pairs.

[Network>Cohesion>Density](#) is a useful tool for calculating the density of whole populations, or of partitions. A typical dialog is shown in figure 8.1.

Figure 8.1. Dialog of Network>Cohesion>Density



In this dialog, we are again examining the Knoke information tie network. We have used an attribute or partition to divide the cases into three sub-populations (governmental agencies, non-governmental generalist, and welfare specialists) so that we can see the amount of connection within and between groups. This is done by creating a separate attribute data file (or a column in such a file), with the same row labels, and scores for each case on the "partitioning" variable. Partitioning is not necessary to calculate density. The results of the analysis are shown in figure 8.2.

Figure 8.2. Density of three sub-populations in Knoke information network

Column	Block	Old Code	Members:									
	1	1	COUN EDUC MAYR									
	2	2	COMM INDU NEWS									
	3	3	WRO UWAY WELF WEST									

Relation: KNOKI

		1	3	5	2	4	7	6	8	9	10
		C	E	M	C	I	N	W	U	W	W
1				1	1	1				1	
3				1	1	1	1	1			1
5		1	1		1	1	1		1	1	1
2		1	1	1		1	1		1	1	
4		1		1	1		1				
7				1	1	1					
6			1				1			1	
8		1		1	1	1	1			1	
9				1	1		1				
10		1	1	1	1	1					

Density / average value within blocks

		1	2	3
		1	2	3
1	1	0.6667	0.8889	0.5000
2	2	0.6667	1.0000	0.1667
3	3	0.5833	0.6667	0.1667

Standard Deviations within blocks

		1	2	3
		1	2	3
1	1	0.4714	0.3143	0.5000
2	2	0.4714	0.0000	0.3727
3	3	0.4930	0.4714	0.3727

After providing a map of the partitioning, a blocked (partitioned) matrix is provided showing the values of the connections between each pair of actors. Next, the within-block densities are presented. The density in the 1,1 block is .6667. That is, of the six possible directed ties among actors 1, 3, and 5, four are actually present (we have ignored the diagonal -- which is the most common approach). We can see that the three sub-populations appear to have some differences. Governmental generalists (block 1) have quite dense in and out ties to one another, and to the other populations; non-government generalists (block 2) have out-ties among themselves and with block 1, and have high densities of in-ties with all three sub-populations. The welfare specialists have high density of information sending to the other two blocks (but not within their

block), and receive more input from governmental than from non-governmental organizations.

The extent to which these simple characterizations of blocks characterize all the individuals within those blocks -- essentially the validity of the blocking -- can be assessed by looking at the standard deviations within the partitions. The standard deviations measure the lack of homogeneity within the partition, or the extent to which the actors vary.

A social structure in which individuals were highly clustered would display a pattern of high densities on the diagonal, and low densities elsewhere.

[table of contents](#)

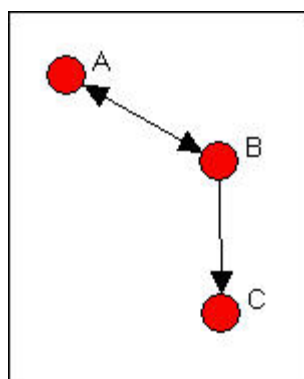
Reciprocity

With symmetric dyadic data, two actors are either connected, or they are not. Density tells up pretty much all there is to know.

With directed data, there are four possible dyadic relationships: A and B are not connected, A sends to B, B sends to A, or A and B send to each other. A common interest in looking at directed dyadic relationships is the extent to which ties are reciprocated. Some theorists feel that there is an equilibrium tendency toward dyadic relationships to be either null or reciprocated, and that asymmetric ties may be unstable. A network that has a predominance of null or reciprocated ties over asymmetric connections may be a more "equal" or "stable" network than one with a predominance of asymmetric connections (which might be more of a hierarchy).

There are (at least) two different approaches to indexing the degree of reciprocity in a population. Consider the very simple network shown in figure 8.3. Actors A and B have reciprocated ties, actors B and C have a non-reciprocated tie, and actors A and C have no tie.

Figure 8.3. Definitions of reciprocity

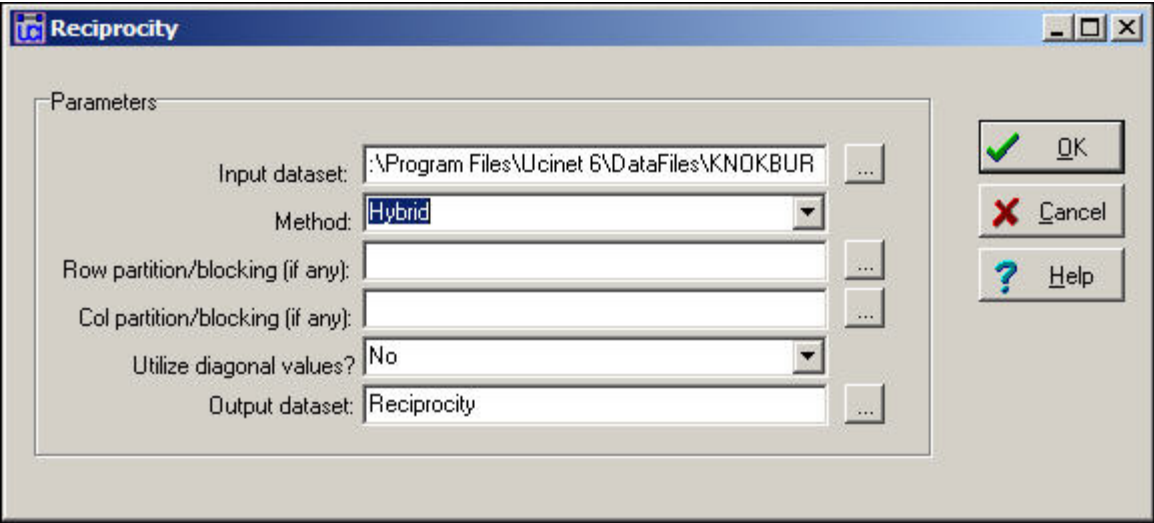


What is the prevalence of reciprocity in this network? One approach is to focus on the dyads, and ask what proportion of pairs have a reciprocated tie between them? This would yield one such tie for three possible pairs (AB, AC, BC), or a reciprocity rate of .333. More commonly, analysts are concerned with the ratio of the number of pairs with a reciprocated tie relative to the number of pairs with any tie. In large populations, usually most actors have no direct ties to most other actors, and it may be more sensible to focus on the degree of reciprocity among pairs that have any ties. In our simple example, this would yield one reciprocated pair divided by two tied pairs, or a reciprocity rate of .500. The method just described is called the dyad method in [Network>Cohesion>Reciprocity](#).

Rather than focusing on actors, we could focus on relations. We could ask, what percentage of all possible

ties (or "arcs" of the directed graph) are parts of reciprocated structures? Here, two such ties (A to B and B to A) are a reciprocated structure among the six possible ties (AB, BA, AC, CA, BC, CA) or a reciprocity of .333. Analysts usually focus, instead, on the number of ties that are involved in reciprocal relations relative to the total number of actual ties (not possible ties). Here, this definition would give us 2 / 3 or .667. This approach is called the arc method in [Network>Cohesion>Reciprocity](#). Here's a typical dialog for using this tool.

Figure 8.4. Dialog for Network>Network Properties>Reciprocity



We've specified the "hybrid" method (the default) which is the same as the dyad approach. Note that it is possible to block or partition the data by some pre-defined attribute (like in the density example above) to examine the degree of reciprocity within and between sub-populations. Figure 8.5 shows the results for the Knoke information network.

Figure 8.5. Reciprocity in the Knoke information network

```
Hybrid Reciprocity: 0.5313

In the hybrid method, the overall reciprocity value is the same as in the dyad-based model.
I.e., Num(Xij>0 and Xji>0)/Num(Xij>0 or Xji>0)
```

We see that, of all all pairs of actors that have any connection, 53% of the pairs have a reciprocated connection. This is neither "high" nor "low" in itself" but does seem to suggest a considerable degree of institutionalized horizontal connection within this organizational population.

The alternative method of "arc" reciprocity (not shown here) yield a result of .6939. That is, of all the relations in the graph, 69% are parts of reciprocated ties.

[table of contents](#)

Transitivity

Small group theorists argue that many of the most interesting and basic questions of social structure arise with regard to triads. Triads allow for a much wider range of possible sets of relations.

With un-directed data, there are four possible types of triadic relations (no ties, one tie, two ties, or all three ties). Counts of the relative prevalence of these four types of relations across all possible triples (that is a "triad census") can give a good sense of the extent to which a population is characterized by "isolation,"

"couples only," "structural holes" (i.e. where one actor is connected to two others, who are not connected to each other), or "clusters." UCINET does not have a routine for conducting triad censuses (see Pajek, which does).

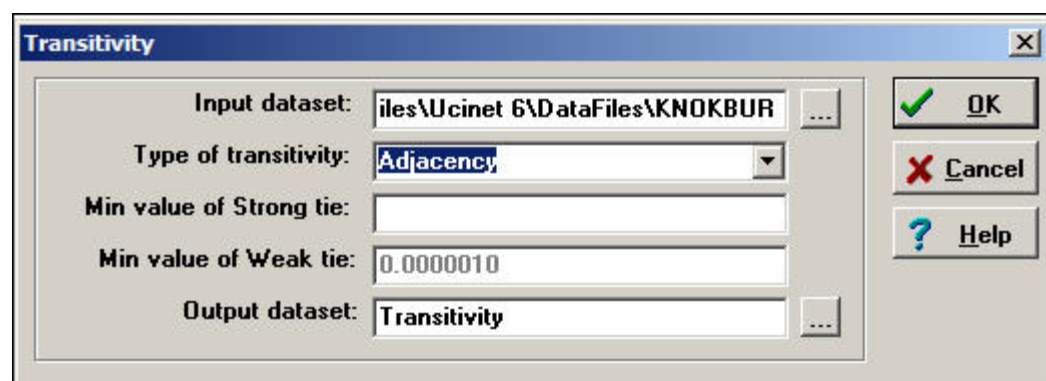
With directed data, there are actually 16 possible types of relations among 3 actors), including relationships that exhibit hierarchy, equality, and the formation of exclusive groups (e.g. where two actors connect, and exclude the third). Thus, small group researchers suggest, all of the really fundamental forms of social relationships can be observed in triads. Because of this interest, we may wish to conduct a "triad census" for each actor, and for the network as a whole (again, see Pajek).

In particular, we may be interested in the proportion of triads that are "transitive" (that is, display a type of balance where, if A directs a tie to B, and B directs a tie to C, then A also directs a tie to C). Such transitive or balanced triads are argued by some theorists to be the "equilibrium" or natural state toward which triadic relationships tend (not all theorists would agree!).

Of the 16 possible types of directed triads, six involve zero, one, or two relations -- and can't display transitivity because there are not enough ties to do so. One type with 3 relations (AB, BC, CB) does not have any ordered triples (AB, BC) and hence can't display transitivity. In three more types of triads, there are ordered triples (AB, BC) but the relation between A and C is not transitive. The remaining types of triads display varying degrees of transitivity.

UCINET does not have extensive algorithms for examining full triad censuses and building more complex models based on them (e.g. balance, clusterability, ranked clusters). A more extended treatment of this approach, with supporting software is available from Pajek. Nonetheless, the [Network>Cohesion>Transitivity](#) algorithms in UCINET offer some interesting and flexible approaches to characterizing the transitivity of triads in populations. A typical dialog is shown in figure 8.6.

Figure 8.6. Dialog of Network>Cohesion>Transitivity



The Knoke information network is a binary, directed graph. For data of this type, the default definition of transitivity (i.e. "Adjacency") is a reasonable approach. This means that we will count the number of times that, if we see AB and BC, we also see AC.

[Network>Cohesion>Transitivity](#) also provides some alternative definitions of what it means for a triad to be transitive which are useful for valued data.

A *strong* transitivity is one in which there are connections AB, BC, and AC, and the connection AC is stronger than the *Min value of Strong tie*. A *weak* transitivity is one in which there are connections AB, BC and AC, and AC; the value of AC is less than the threshold for a strong tie, but greater than the threshold *Min value of Weak tie*.

Two other methods are also available. A *Euclidean* transitivity is defined as a case where AB, BC, and AC are present, and AC has a value less than the sum of AB + BC. A *Stochastic* transitivity is defined as the case where AB, BC, and AC are present, and AC is less than the produce AB*BC.

Figure 8.7. Transitivity results for Knoke information network

TRANSITIVITY	
Type of transitivity:	ADJACENCY
Input dataset:	C:\Program Files\Ucinet 6\DataFiles\KNOKBUR
Relation:	KNOKI
Number of non-vacuous transitive ordered triples: 146	
Number of triples of all kinds: 720	
Number of triples in which i-->j and j-->k: 217	
Percentage of all ordered triples: 20.28%	
Transitivity: % of ordered triples in which i-->j and j-->k that are transitive: 67.28%	

After performing a census of all possible triads, [Network>Cohesion>Transitivity](#) reports that it finds 146 transitive (directed) triples. That is, there are 146 cases where, if AB and BC are present, then AC is also present. There are a number of different ways in which we could try to norm this count so that it becomes more meaningful. One approach is to divide the number of transitive triads by the total number of triads of all kinds (720). This shows that 20.28% of all triads are transitive. Perhaps more meaningful is to norm the number of transitive triads by the number of cases where a single link could complete the triad. That is, norm the number of {AB, BC, AC} triads by the number of {AB, BC, anything} triads. Seen in this way, about 2/3 or all relations that could easily be transitive, actually are.

[table of contents](#)

Clustering

Watts (1999) and many others have noted that in large, real-world networks (of all kinds of things) there is often a structural pattern that seems somewhat paradoxical.

On one hand, in many large networks (like, for example, the Internet) the average geodesic distance between any two nodes is relatively short. The "6-degrees" of distance phenomenon is an example of this. So, most of the nodes in even very large networks may be fairly close to one another. The average distance between pairs of actors in large empirical networks are often much shorter than in random graphs of the same size.

On the other hand, most actors live in local neighborhoods where most others are also connected to one another. That is, in most large networks, a very large proportion of the total number of ties are highly "clustered" into local neighborhoods. That is, the density in local neighborhoods of large graphs tend to be much higher than we would expect for a random graph of the same size.

Most of the people we know may also know one another -- seeming to locate us in a very narrow social world. Yet, at the same time, we can be at quite short distances to vast numbers of people that we don't know at all. The "small world" phenomena -- a combination of short average path lengths over the entire graph, coupled with a strong degree of "clique-like" local neighborhoods -- seems to have evolved independently in many large networks.

We've already discussed one part of this phenomenon. The average geodesic distance between all actors in

a graph gets at the idea of how close actors are together. The other part of the phenomenon is the tendency towards dense local neighborhoods, or what is now thought of as "clustering."

One common way of measuring the extent to which a graph displays clustering is to examine the local neighborhood of an actor (that is, all the actors who are directly connected to ego), and to calculate the density in this neighborhood (leaving out ego). After doing this for all actors in the whole network, we can characterize the degree of clustering as an average of all the neighborhoods.

Figure 8.8 shows the output of *Network>Cohesion>Clustering Coefficient* as applied to the Knoke information network.

Figure 8.8. Network>Cohesion>Clustering Coefficient of Knoke information network

Input dataset:	C:\Program Files\Ucinet 6\
Relation:	KNOKI

Overall graph clustering coefficient:	0.607
Weighted Overall graph clustering coefficient:	0.599

Two alternative measures are presented. The "overall" graph clustering coefficient is simply the average of the densities of the neighborhoods of all of the actors. The "weighted" version gives weight to the neighborhood densities proportional to their size; that is, actors with larger neighborhoods get more weight in computing the average density. Since larger graphs are generally (but not necessarily) less dense than smaller ones, the weighted average neighborhood density (or clustering coefficient) is usually less than the un-weighted version. In our example, we see that all of the actors are surrounded by local neighborhoods that are fairly dense -- our organizations can be seen as embedded in dense local neighborhoods to a fairly high degree. Lest we over-interpret, we must remember that the overall density of the entire graph in this population is rather high (.54). So, the density of local neighborhoods is not really much higher than the density of the whole graph. In assessing the degree of clustering, it is usually wise to compare the cluster coefficient to the overall density.

We can also examine the densities of the neighborhoods of each actor, as is shown in figure 8.9.

Figure 8.9. Node level clustering coefficients for Knoke information network

Node Clustering Coefficients		
	1	2
	Clus C	nPairs
1	0.667	21.000
2	0.536	28.000
3	0.567	15.000
4	0.733	15.000
5	0.518	28.000
6	0.333	3.000
7	0.514	36.000
8	0.800	15.000
9	0.600	15.000
10	0.800	10.000

The sizes of each actor's neighborhood is reflected in the number of pairs of actors in it. Actor 6, for example has three neighbors, and hence three possible ties. Of these, only one is present -- so actor 6 is not highly clustered. Actor 8, on the other hand, is in a slightly larger neighborhood (6 neighbors, and hence 15 pairs of neighbors), but 80% of all the possible ties among these neighbors are present. Actors 8 and 10 are embedded in highly clustered neighborhoods.

[table of contents](#)

Group-external and group-internal ties

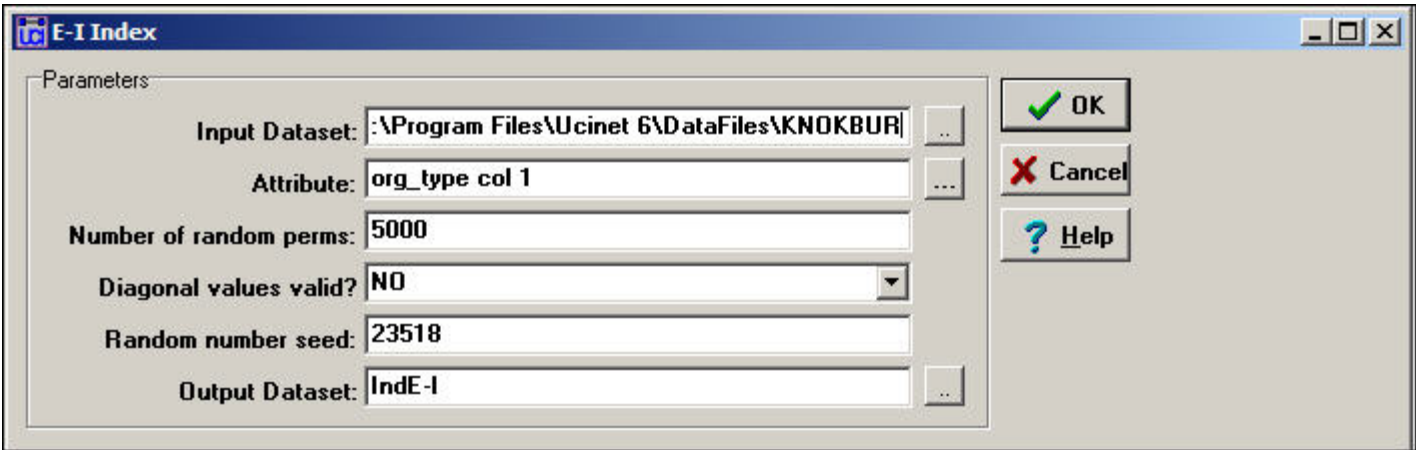
Actors may be embedded in macro-structures, as well as in dyads, triads, and neighborhoods. Some macro-structures are social agents (like voluntary and formal organizations); some macro-structures are categorical units (like gender and ethnic groups). To understand the "texture" of the "social fabric" we might want to index the extent to which these macro-structures "cluster" the interaction patterns of individuals who fall within them.

Krackhardt and Stern (1988) developed a very simple and useful measure of the group embedding based on comparing the numbers of ties within groups and between groups. The E-I (external - internal) index takes the number of ties of group members to outsiders, subtracts the number of ties to other group members, and divides by the total number of ties. The resulting index ranges from -1 (all ties are internal to the group) to +1 (all ties are external to the group). Since this measure is concerned with any connection between members, the directions of ties are ignored (i.e. either a out-tie or an in-tie constitutes a tie between two actors).

The E-I index can be applied at three levels: the entire population, each group, and each individual. That is, the network as a whole (all the groups) can be characterized in terms of the bounded-ness and closure of its sub-populations. We can also examine variation across the groups in their degree of closure; and, each individual can be seen as more or less embedded in their group.

Here's a sample of the dialog with [Network>Cohesion>E-I Index](#) in which we examine the Knoke information network that has been partitioned according to the attribute of organizational type (group 1 = governmental generalists, group 2 = non-governmental generalists, group 3 = welfare specialists).

Figure 8.10. Dialog of Network>Cohesion>E-I Index



The range of possible values of the E-I index is restricted by the number of groups, relative group sizes, and total number of ties in a graph. Often this range restriction is quite severe, so it is important to re-scale the coefficient to range between the maximum possible degree of "external-ness" (+1) and the maximum possible degree of "internal-ness." As Blau and others have noted, the relative sizes of sub-populations have dramatic consequences for the degree of internal and external contacts, even when individuals may choose contacts at random.

To assess whether a give E-I index value is significantly different that what would be expected by random mixing (i.e. no preference for within or without group ties by group members), a permutation test is performed by [Network>Cohesion>E-I Index](#). A large number of trials are run in which the blocking of groups is maintained, and the overall density of ties is maintained, but the actual ties are randomly distributed. From a

large number of trials (the default is 5000), a sampling distribution of the numbers of internal and external ties -- under the assumption that ties are randomly distributed -- can be calculated. This sampling distribution can then be used to assess the frequency with which the observed result would occur by sampling from a population in which ties were randomly distributed.

Let's look first at the results for the graph as a whole, in figure 8.11.

Figure 8.11. E-I index output for the Knoke information network - whole network

Density matrix								
		1	2	3				
		1	2	3				
	1	1	0.667	1.000	0.667			
	2	2	1.000	1.000	0.667			
	3	3	0.667	0.667	0.333			
64 ties.								
Whole Network Results								
		1	2	3	4			
		Freq	Pct	Possib	Densit			
1	Internal	14.000	0.219	24.000	0.583			
2	External	50.000	0.781	66.000	0.758			
3	E-I	36.000	0.563	42.000	0.467			
Max possible external ties: 66.000								
Max possible internal ties: 24.000								
E-I Index: 0.563								
Expected value for E-I index is: 0.467								
Max possible E-I given density & group sizes: 1.000								
Min possible E-I given density & group sizes: 0.250								
Re-scaled E-I index: -0.167								
Permutation Test								
Number of iterations = 5000								
		1	2	3	4	5	6	7
		Obs	Min	Avg	Max	SD	P >= Ob	P <= Ob
1	Internal	0.219	0.625	0.733	0.844	0.039	1.000	0.000
2	External	0.781	0.156	0.267	0.375	0.039	0.000	1.000
3	E-I	0.563	0.250	0.467	0.688	0.078	0.203	0.953

The observed block densities are presented first. Since any tie (in or out) is regarded as a tie, the densities in this example are quite high. The densities off the main diagonal (out-group ties) appear to be slightly more prevalent than the densities on the main diagonal (in-group ties).

Next, we see the numbers of internal ties (14, or 22%) and external ties (50, or 78%) that yield a raw (not rescaled) E-I index of +.563. That is, a preponderance of external over internal ties for the graph as a whole. Also shown are the maximum possible numbers of internal and external ties given the group sizes and density. Note that, due to these constraints, the result of a preponderance of external ties is not unexpected -- under a random distribution, the E-I index would be expected to have a value of .467, which is not very much different from the observed value.

We see that, given the group sizes and density of the graph, the maximum possible value of the index (1.0) and its minimum value (+.25) are both positive. If we re-scale the observed value of the E-I index (.563) to

fall into this range, we obtain a re-scaled index value of $-.167$. This suggests, that, given the demographic constraints and overall density, there is a very modest tendency toward group closure.

The last portion of the results gives the values of the permutation-based sampling distribution. Most important here is the standard deviation of the sampling distribution of the index, or its standard error ($.078$). This suggests that the value of the raw index is expected to vary by this much from trial to trial (on the average) just by chance. Given this result, we can compare the observed value in our sample ($.563$) to the expected value ($.467$) relative to the standard error. The observed difference of about $.10$ could occur fairly frequently just by sampling variability ($p = .203$). Most analysts would not reject the null hypothesis that the deviation from randomness was not "significant." That is, we cannot be confident that the observed mild bias toward group closure is not random variation.

The E-I index can also be calculated for each group and for each individual. These index numbers describe the tendencies toward group closure of each of the groups, and the propensity of each individual to have ties within their group. Figure 8.12 displays the results.

Figure 8.12. E-I index output for the Knoke information network - groups and individuals

Group level E-I Index					
		1	2	3	4
		Intern	Extern	Total	E-I
1	1	4.000	17.000	21.000	0.619
2	2	6.000	17.000	23.000	0.478
3	3	4.000	16.000	20.000	0.600

Individual Level E-I Index					
		1	2	3	4
		Inter	Exter	Total	E-I
1	1.000	6.000	7.000	0.714	
2	2.000	6.000	8.000	0.500	
3	1.000	5.000	6.000	0.667	
4	2.000	4.000	6.000	0.333	
5	2.000	6.000	8.000	0.500	
6	1.000	2.000	3.000	0.333	
7	2.000	7.000	9.000	0.556	
8	1.000	5.000	6.000	0.667	
9	2.000	4.000	6.000	0.333	
10	0.000	5.000	5.000	1.000	

The first panel of figure 8.12 shows the raw counts of ties within and without each of the three types of organizations, and the E-I index for each group. Governmental generalists (group 2) appear to be somewhat more likely to have out-group ties than either of the other sub-populations. The relatively small difference, though, should be treated with considerable caution given the sampling variability (we cannot directly apply the standard error estimate for the whole graph to the results for sub-populations or individuals, but they are suggestive). We should also note that the E-I results for groups and individuals are in "raw" form, and not "rescaled."

There is considerable variability across individuals in their propensity to in-group ties, as can be seen in the last panel of the results. Several actors (4, 6, 9) tend toward closure -- having a preponderance of ties within their own group; a couple others (10, 1) tend toward a preponderance of ties outside their groups.

[table of contents](#)

Embedding of actors in dyads, triads, neighborhoods, clusters, and groups are all ways in which the social structure of a population may display "texture." All of these forms of embedding structures speak to the issue of the "horizontal differentiation" of the population -- separate, but not necessarily ranked or unequal groupings.

A very common form of embedding of actors in structures, though, does involve unequal rankings. Hierarchies, in which individuals or sub-populations are not only differentiated, but also ranked, are extremely common in social life. The degree of hierarchy in a population speaks to the issue of "vertical differentiation."

While we all have an intuitive sense of what it means for a structure to be a hierarchy. Most would agree that structures can be "more or less" hierarchical. It is necessary to be quite precise about the meaning of the term if we are going to build indexes to measure the degree of hierarchy.

Krackhardt (1994) provided an elegant definition of the meaning of hierarchy, and developed measures of each of the four component dimensions of the concept that he identified. Krackhardt defines a pure, "ideal typical" hierarchy as an "out-tree" graph. An out-tree graph is a directed graph in which all points are connected, and all but one node (the "boss") has an in-degree of one. This means that all actors in the graph (except the ultimate "boss") have a single superior node. The simplest "hierarchy" is a directed line graph A to B to C to D... More complex hierarchies may have wider, and varying "spans of control" (out-degrees of points).

This very simple definition of the pure type of hierarchy can be deconstructed into four individually necessary and jointly sufficient conditions. Krackhardt develops index numbers to assess the extent to which each of the four dimensions deviates from the pure ideal type of an out-tree, and hence develops four measures of the extent to which a given structure resembles the ideal typical hierarchy.

1) Connectedness: To be a pure out-tree, a graph must be connected into a single component -- all actors are embedded in the same structure. We can measure the extent to which this is not true by looking at the ratio of the number of pairs in the directed graph that are reachable relative to the number of ordered pairs. That is, what proportion of actors cannot be reached by other actors? Where a graph has multiple components -- multiple un-connected sub-populations -- the proportion not reachable can be high. If all the actors are connected in the same component, if there is a "unitary" structure, the graph is more hierarchical.

2) Hierarchy: To be a pure out-tree, there can be no reciprocated ties. Reciprocal relations between two actors imply equal status, and this denies pure hierarchy. We can assess the degree of deviation from pure hierarchy by counting the number of pairs that have reciprocated ties relative to the number of pairs where there is any tie; that is, what proportion of all tied pairs have reciprocated ties.

3) Efficiency: To be a pure out-tree each node must have an in-degree of one. That is, each actor (except the ultimate boss) has a single boss. This aspect of the idea type is termed "efficiency" because structures with multiple bosses have un-necessary redundant communication of orders from superiors to subordinates. The amount of deviation from this aspect of the pure out-tree can be measured by counting the difference between the actual number of links (minus 1, since the ultimate boss has no boss) and the maximum possible number of links. The bigger the difference, the greater the inefficiency. This dimension then measures the extent to which actors have a "single boss."

4) Least upper bound (LUB): To be a pure out-tree, each pair of actors (except pairs formed between the ultimate boss and others) must have an actor that directs ties to both -- that is, command must be unified. The deviation of a graph from this condition can be measured by counting the numbers of pairs of actors that do not have a common boss relative to the number of pairs that could (which depends on the number of

actors and the span of control of the ultimate boss).

The [Network>Cohesion>Krackhardt GTD](#) algorithms calculate indexes of each of the four dimensions, where higher scores indicate greater hierarchy. Figure 8.13 shows the results for the Knoke information network.

Figure 8.13. Output of Network>Network Properties>Krackhardt GDT for Knoke information network

Krackhardt GTD Measures		
		1
1	Connectedness	1.0000
2	Hierarchy	0.0000
3	Efficiency	0.3611
4	LUB	1.2500

The information network does form a single component, as there is at least one actor that can reach all others. So, the first dimension of pure hierarchy -- that all the actors be embedded in a single structure -- is satisfied. The ties in the information exchange network, though are very likely to be reciprocal (at least insofar as they can be, given the limitations of the density). There are a number of nodes that receive information from multiple others, so the network is not "efficient." The least upper bound measure (the extent to which all actors have a boss in common) reports a value of 1.25, which would appear to be out of range and, frankly, is a puzzle.

[table of contents](#)

Summary

This chapter and the next are concerned with the ways in which networks display "structure" or deviation from random connection. In the current chapter, we've approached the same issue of structuring from the "top-down" by looking at patterns of macro-structure in which individuals are embedded in non-random ways. Individuals are embedded (usually simultaneously) in dyads, triads, face-to-face local groups of neighbors, and larger organizational and categorical social structures. The tools in the current chapter provide some ways of examining the "texture" of the structuring of the whole population.

In the next chapter we will focus on the same issue of connection and structure from the "bottom-up." That is, we'll look at structure from the point of view of the individual "ego."

Taken together, the approaches in chapters 8 and 9 illustrate, again, the "duality" of social structure in which individuals make social structures, but do so within a matrix of constraints and opportunities imposed by larger patterns.

[table of contents](#)
[table of contents of the book](#)

Introduction to social network methods

9. Ego networks

This page is part of an on-line text by [Robert A. Hanneman](#) ([Department of Sociology, University of California, Riverside](#)) and Mark Riddle ([Department of Sociology, University of Northern Colorado](#)). Feel free to use and distribute this textbook, with citation. Your comments and suggestions are very welcome. [Send me e-mail.](#)

Contents of chapter 9: Ego networks

- [Introduction](#)
 - [Ego network data](#)
 - [Ego network density](#)
 - [Structural holes](#)
 - [Brokerage](#)
 - [Summary](#)
-

Introduction

In the previous chapter we looked at the idea of the amount of "embedding" in whole networks -- loosely: the extent to which actors find themselves in social structures characterized by dense, reciprocal, transitive, strong ties. The main theme was to understand and index the extent and nature of the pattern of "constraint" on actors that results from the way that they are connected to others. These approaches may tell us some interesting things about the entire population and its sub-populations; but, they don't tell us very much about the opportunities and constraints facing individuals.

If we want to understand variation in the behavior of individuals, we need to take a closer look at their local circumstances. Describing and indexing the variation across individuals in the way they are embedded in "local" social structures is the goal of the analysis of ego networks.

We need some definitions.

"Ego" is an individual "focal" node. A network has as many egos as it has nodes. Egos can be persons, groups, organizations, or whole societies.

"Neighborhood" is the collection of ego and all nodes to whom ego has a connection at some path length. In social network analysis, the "neighborhood" is almost always one-step; that is, it includes only ego and actors that are directly adjacent. The neighborhood also includes all of the ties among all of the actors to whom ego has a direct connection. The boundaries of ego networks are defined in terms of neighborhoods.

"N-step neighborhood" expands the definition of the size of ego's neighborhood by including all nodes to whom ego has a connection at a path length of N, and all the connections among all of these actors. Neighborhoods of greater path length than 1 (i.e. egos adjacent nodes) are rarely used in social network analysis. When we use the term neighborhood here, we mean the one-step neighborhood.

"In" and "Out" and other kinds of neighborhoods. Most of the analysis of ego networks uses simple graphs (i.e. graphs that are symmetric, and show only connection/not, not direction). If we are working with a directed graph, it is possible to define different kinds of ego-neighborhoods. An "out" neighborhood would include all the actors to whom ties are directed from ego. An "in" neighborhood would include all the actors who sent ties directly to ego. We might want to define a neighborhood of only those actors to whom ego had reciprocated ties. There isn't a single "right" way to define an ego neighborhood for every research question.

"Strong and weak tie neighborhoods." Most analysis of ego networks uses binary data -- two actors are connected

or they aren't, and this defines the ego neighborhood. But if we have measured the strength of the relation between two actors, and even its valence (positive or negative), we need to make choices about when we are going to decide that another actor is ego's neighbor. With ties that are measured as strengths or probabilities, a reasonable approach is to define a cut-off value (or, better, explore several reasonable alternatives). Where the information about ties includes information about positive/negative, the most common approach is to analyze the positive tie neighborhood and the negative tie neighborhood separately.

[table of contents](#)

Ego network data

Ego network data commonly arise in two ways:

Surveys may be used to collect information on ego networks. We can ask each research subject to identify all of the actors to whom they have a connection, and to report to us (as an informant) what the ties are among these other actors. Alternatively, we could use a two-stage snowball method; first ask ego to identify others to whom ego has a tie, then ask each of those identified about their ties to each of the others identified.

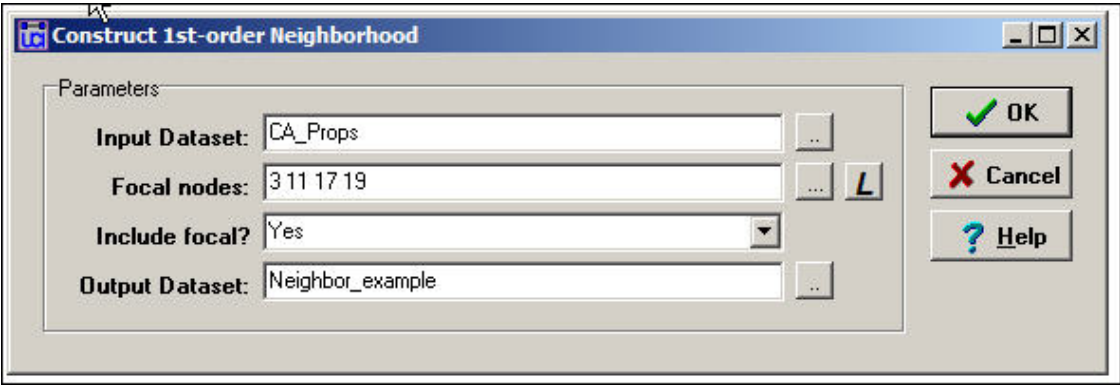
Data collected in this way cannot directly inform us about the overall embeddedness of the networks in a population, but it can give us information on the prevalence of various kinds of ego networks in even very large populations. When data are collected this way, we essentially have a data structure that is composed of a collection of networks. As the actors in each network are likely to be different people, the networks need to be treated as separate actor-by-actor matrices stored as different data sets (i.e. it isn't a good idea to "stack" the multiple networks in the same data file, because the multiple matrices do not represent multiple relations among the same set of actors).

A modification of the survey method can give rise to a multi-plex data structure (that is, a "stack" of actor-by-actor matrices of equal dimension). If we ask each ego to characterize their relation with the occupants of social roles (or a particular occupant of a role), and to also report on the relations among occupants of those roles, we can build "conformable" matrices for each ego. For example, suppose that we asked a number of egos: "do you have a male friend or friends in your classroom?" "Do you have a female friend or friends in your classroom?" and "Are your male friends, friends of your female friends?" The resulting data for each ego would have three nodes (ego, "male friends," "female friends") and the ties among them. Since each ego's matrix would have the same nodes (in the sense of social roles, but not individuals) they could be treated as a type of multi-plex data that we will discuss more later on.

The second major way in which ego network data arise is by "extracting" them from regular complete network data. The [Data>Extract](#) approach can be used to select a single actor and their ties, but would not include the ties among the "alters." The [Data>Subgraphs from partitions](#) approach could be used if we had previously identified the members of a particular ego neighborhood, and stored this as an attribute vector.

More commonly, though, we would want to extract multiple, or even all of the ego networks from a full network to be stored as separate files. For this task, the [Data>Egonet](#) tool is ideal. Here is an example of the dialog for using the tool:

Figure 9.1. Dialog for Data>Egonet



Here we are focusing on ballot proposition campaigns in California that are connected by having donors in common (i.e. CA_Props is a proposition-by-proposition valued matrix). We've said that we want to extract a network that includes the 3rd, 11th, 17th, and 19th rows/columns, and all the nodes that are connected to any of these actors. More commonly, we might select a single "ego." The list of focal nodes can be provided either as an attribute file, by typing in the list of row numbers, or by selecting the node labels of the desired actors.

A picture of part of the resulting data, stored as a new file called "Neighbor_example" is shown in figure 9.2.

Figure 9.2. (Partial) output of Data>Egonet

CONSTRUCT 1ST-ORDER NEIGHBORHOOD															
Input dataset:		C:\Documents and Settings\hanneman\													
focal nodes:		3 11 17 19													
Include focal nodes?		YES													
Output dataset:		Neighbor_example													
		3	11	17	19	4	5	10	14	15	16	18	20	21	23
		P13	P28	P36	P38	P14	P15	P26	P33	P34	P35	P37	P39	P40	P42
3	P13	9	-1	1	-2	1	2	2	1	3	2	1	4	3	1
11	P28	-1	8	-1	1	-2	-1	-2	-1	-1	0	1	-3	0	3
17	P36	1	-1	4	-1	1	1	1	1	1	0	0	0	0	-1
19	P38	-2	1	-1	13	-2	-1	-6	-4	-5	1	1	-6	0	2
4	P14	1	-2	1	-2	3	1	2	1	1	0	0	2	0	0
5	P15	2	-1	1	-1	1	3	1	1	1	0	1	1	1	0
10	P26	2	-2	1	-6	2	1	18	3	3	2	-1	14	1	-1
14	P33	1	-1	1	-4	1	1	3	5	4	-1	-1	2	0	-2
15	P34	3	-1	1	-5	1	1	3	4	9	-1	0	3	1	-2
16	P35	2	0	0	1	0	0	2	-1	-1	7	0	2	0	1
18	P37	1	1	0	1	0	1	-1	-1	0	0	6	0	1	1
20	P39	4	-3	0	-6	2	1	14	2	3	2	0	25	3	1
21	P40	3	0	0	0	0	1	1	0	1	0	1	3	10	2
23	P42	1	3	-1	2	0	0	-1	-2	-2	1	1	1	2	10
24	P45	4	-6	1	-6	3	3	7	5	6	-1	0	11	4	1
25	P46	2	-3	1	-6	2	1	6	3	4	0	-1	7	1	-2
26	P47	4	-2	0	-5	1	1	9	2	3	3	0	13	3	1
27	P49	1	-4	0	0	2	1	3	0	0	0	1	6	4	2
28	P50	2	0	0	0	0	1	1	0	0	0	1	2	7	2
30	P52	1	-3	1	-1	1	1	2	2	2	0	-1	1	0	-2
38	P62	2	-1	0	2	1	0	5	-1	-1	2	1	6	2	3
39	P63	4	-4	1	-4	2	1	2	2	4	-1	0	4	1	-3

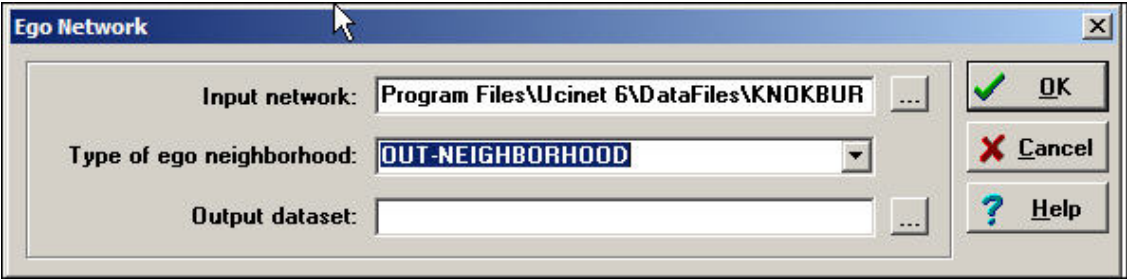
Extracting sub-graphs, based on a focal actor or set of actors (e.g. "elites") can be a very useful way of looking at a part of a whole network, or the condition of an individual actor. The [Data>Egonet](#) tool is helpful for creating data sets that are good for graphing and separate analysis -- particularly when the networks in which the focal actor/actors are embedded are quite large.

It is not necessary, however, to create separate ego-network datasets for each actor to be analyzed. The approaches to analysis that we'll review below generate output for the first-order ego network of every node in a dataset. For small datasets, there is often no need to extract separate ego networks.

Ego network density

There are quite a few characteristics of the ego-neighborhoods of actors that may be of interest. The [Network>Ego networks>Density](#) tools in UCINET calculate a substantial number of indexes that describe aspects of the neighborhood of each ego in a data set. Here is an example of the dialog, applied to the Knoke information exchange data (these are binary, directed connections).

Figure 9.3. Dialog for Network>Ego networks>Density



In this example, we've decided to examine "out neighborhoods" (in neighborhoods or undirected neighborhoods can also be selected). We've elected not to save the output as a dataset (if you wanted to do further analysis, or treat ego network descriptive statistics as node attributes, you might want to save the results as a file for use in other routines or Netdraw). Here are the results:

Figure 9.4 Ego network density output for Knoke information out-neighborhoods

Density Measures														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	Size	Ties	Pairs	Densit	AvgDis	Diamet	nWeakC	pWeakC	2StepR	ReachE	Broker	nBroke	EgoBet	nEgoBe
1	4.00	11.00	12.00	91.67	1.08	2.00	1.00	25.00	100.00	29.03	0.50	0.04	0.00	0.00
2	7.00	24.00	42.00	57.14	1.43	2.00	1.00	14.29	100.00	18.75	9.00	0.21	8.17	19.44
3	6.00	17.00	30.00	56.67			1.00	16.67	100.00	23.08	6.50	0.22	8.25	27.50
4	4.00	11.00	12.00	91.67	1.08	2.00	1.00	25.00	100.00	28.13	0.50	0.04	0.33	2.78
5	8.00	29.00	56.00	51.79	1.57	3.00	1.00	12.50	100.00	16.98	13.50	0.24	14.67	26.19
6	3.00	2.00	6.00	33.33			1.00	33.33	100.00	42.86	2.00	0.33	1.00	16.67
7	3.00	6.00	6.00	100.00	1.00	1.00	1.00	33.33	88.89	36.36	0.00	0.00	0.00	0.00
8	6.00	24.00	30.00	80.00	1.20	2.00	1.00	16.67	100.00	20.45	3.00	0.10	0.00	0.00
9	3.00	6.00	6.00	100.00	1.00	1.00	1.00	33.33	100.00	36.00	0.00	0.00	0.00	0.00
10	5.00	16.00	20.00	80.00	1.20	2.00	1.00	20.00	100.00	23.68	2.00	0.10	0.33	1.67
1. Size. Size of ego network.														
2. Ties. Number of directed ties.														
3. Pairs. Number of ordered pairs.														
4. Density. Ties divided by Pairs.														
5. AvgDist. Average geodesic distance.														
6. Diameter. Longest distance in egonet.														
7. nWeakComp. Number of weak components.														
8. pWeakComp. NWeakComp divided by Size.														
9. 2StepReach. # of nodes within 2 links of ego.														
10. ReachEffic. 2StepReach divided Size.														
11. Broker. # of pairs not directly connected.														
12. Normalized Broker. Broker divided by number of pairs.														
13. Ego Betweenness. Betweenness of ego in own network.														
14. Normalized Ego Betweenness. Betweenness of ego in own network.														

There's a lot of information here, and we should make a few comments.

Note that there is a line of data for each of the 10 organizations in the data set. Each line describes the one-step ego neighborhood of a particular actor. Of course, many of the actors are members of many of the neighborhoods -- so each actor may be involved in many lines of data.

Size of ego network is the number of nodes that one-step out neighbors of ego, plus ego itself. Actor 5 has the

largest ego network, actors 6, 7, and 9 have the smallest networks.

Number of directed ties is the number of connections among all the nodes in the ego network. Among the four actors in ego 1's network, there are 11 ties.

Number of ordered pairs is the number of possible directed ties in each ego network. In node 1's network there are four actors, so there are 4×3 possible directed ties.

Density is, as the output says, the number of ties divided by the number of pairs. That is, what percentage of all possible ties in each ego network are actually present? Note that actor 7 and 9 live in neighborhoods where all actors send information to all other actors; they are embedded in very dense local structures. The welfare rights organization (node 6) lives in a small world where the members are not tightly connected. This kind of difference in the constraints and opportunities facing actors in their local neighborhoods may be very consequential.

Average geodesic distance is the mean of the shortest path lengths among all connected pairs in the ego network. Where everyone is directly connected to everyone (e.g. node 7 and 9) this distance is one. In our example, the largest average path length for connected neighbors is for actor 5 (average distances among members of the neighborhood is 1.57).

Diameter of an ego network is the length of the longest path between connected actors (just as it is for any network). The idea of a network diameter, is to index the span or extensiveness of the network -- how far apart are the two furthest actors. In the current example, they are not very far apart in the ego networks of most actors.

In addition to these fairly basic and reasonably straight-forward measures, the output provides some more exotic measures that get at some quite interesting ideas about ego neighborhoods that have been developed by a number of social network researchers.

Number of weak components. A weak component is the largest number of actors who are connected, disregarding the direction of the ties (a strong component pays attention to the direction of the ties for directed data). If ego was connected to A and B (who are connected to one another), and ego is connected to C and D (who are connected to one another), but A and B are not connected in any way to C and D (except by way of everyone being connected to ego) then there would be two "weak components" in ego's neighborhood. In our example, there are no such cases -- each ego is embedded in a single component neighborhood. That is, there are no cases where ego is the only connection between otherwise dis-joint sets of actors.

Number of weak components divided by size. The likelihood that there would be more than one weak components in ego's neighborhood would be a function of neighborhood size if connections were random. So, to get a sense of whether ego's role in connecting components is "unexpected" given the size of their network, it is useful to normalize the count of components by size. In our example, since there are no cases of multiple components, this is a pretty meaningless exercise.

Two-step reach goes beyond ego's one-step neighborhood to report the percentage of all actors in the whole network that are within two directed steps of ego. In our example, only node 7 cannot get a message to all other actors within "friend-of-a-friend" distance.

Reach efficiency (two-step reach divided by size) norms the two-step reach by dividing it by size. The idea here is: how much (non-redundant) secondary contact to I get for each unit of primary contact? If reach efficiency is high, then I am getting a lot of "bang for my buck" in reaching a wider network for each unit of effort invested in maintaining a primary contact. If my neighbors, on the average, have few contacts that I don't have, I have low efficiency.

Brokerage (number of pairs not directly connected). The idea of brokerage (more on this, below) is that ego is the "go-between" for pairs of other actors. In an ego network, ego is connected to every other actor (by definition). If these others are not connected directly to one another, ego may be a "broker" ego falls on a the paths between the others. One item of interest is simply how much potential for brokerage there is for each actor (how many times

pairs of neighbors in ego's network are not directly connected). In our example, actor number 5, who is connected to almost everyone, is in a position to broker many connections.

Normalized brokerage (brokerage divided by number of pairs) assesses the extent to which ego's role is that of broker. One can be in a brokering position a number of times, but this is a small percentage of the total possible connections in a network (e.g. the network is large). Given the large size of actor 5's network, the relative frequency with which actor 5 plays the broker role is not so exceptional.

Betweenness is an aspect of the larger concept of "centrality." A later chapter provides a more in-depth treatment of the concept and its application to whole networks. For the moment, though, it's pretty easy to get the basic idea. Ego is "between" two other actors if ego lies on the shortest directed path from one to the other. The ego betweenness measure indexes the percentage of all geodesic paths from neighbor to neighbor that pass through ego.

Normalized Betweenness compares the actual betweenness of ego to the maximum possible betweenness in neighborhood of the size and connectivity of ego's. The "maximum" value for betweenness would be achieved where ego is the center of a "star" network; that is, no neighbors communicate directly with one another, and all directed communications between pairs of neighbors go through ego.

The ideas of "brokerage" and "betweenness" are slightly differing ways of indexing just how "central" or "powerful" ego is within their own neighborhood. This aspect of how an actor's embedding may provide them with strategic advantage has received a great deal of attention. The next two sections, on "structural holes" and "brokerage" elaborate on ways of looking at positional opportunity and constraint of individual actors.

[table of contents](#)

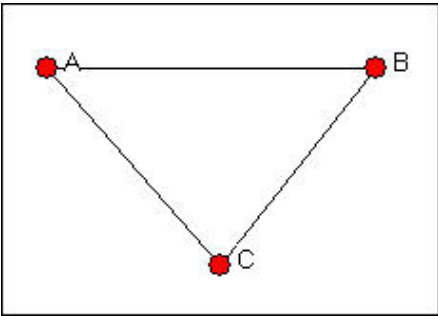
Structural holes

In several important works, Ronald Burt coined and popularized the term "structural holes" to refer to some very important aspects of positional advantage/disadvantage of individuals that result from how they are embedded in neighborhoods. Burt's formalization of these ideas, and his development of a number of measures (including the computer program *Structure*, that provides these measures and other tools) has facilitated a great deal of further thinking about how and why the ways that an actor is connected affect their constraints and opportunities, and hence their behavior.

The basic idea is simple, as good ideas often are.

Imagine a network of three actors (A, B, and C), in which each is connected to each of the others as in figure 9.5.

Figure 9.5. Three actor network with no structural holes

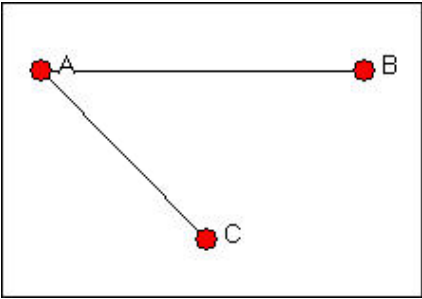


Let's focus on actor A (of course, in this case, the situations of B and C are identical in this particular network). Suppose that actor A wanted to influence or exchange with another actor. Assume that both B and C may have some interest in interacting or exchanging, as well. Actor A will not be in a strong bargaining position in this network, because both of A's potential exchange partners (B and C) have alternatives to treating with A; they could

isolate A, and exchange with one another.

Now imagine that we open a "structural hole" between actors B and C, as in figure 9.6. That is, a relation or tie is "absent" such that B and C cannot exchange (perhaps they are not aware of one another, or there are very high transaction costs involved in forming a tie).

Figure 9.6. Three actor network with a structural hole

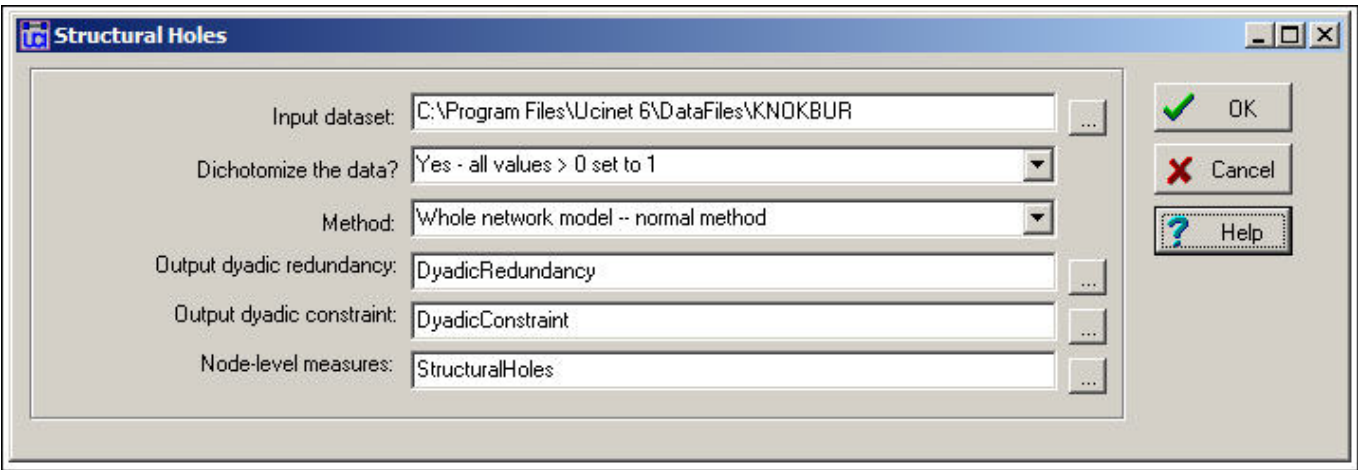


In this situation, actor A has an advantaged position as a direct result of the "structural hole" between actors B and C. Actor A has two alternative exchange partners; actors B and C have only one choice, if they choose to (or must) enter into an exchange.

Real networks, of course, usually have more actors. But, as networks grow in size, they tend to become less dense (how many relations can each actor support?). As density decreases, more "structural holes" are likely to open in the "social fabric." These holes, and how and where they are distributed can be a source of inequality (in both the strict mathematical sense and the sociological sense) among actors embedded in networks.

[Network>Ego Networks>Structural Holes](#) examines the position of each actor in their neighborhood for the presence of structural holes. A number of measures (most proposed by Burt) that describe various aspects of the advantage or disadvantage of the actor are also computed. Figure 9.7 shows a typical dialog box; we're looking at the Knoke information network again.

Figure 9.7. Network>Ego Networks>Structural Holes dialog



Measures related to structural holes can be computed on both valued and binary data. The normal practice in sociological research has been to use binary (a relation is present or not). Interpretation of the measures becomes quite difficult with valued data (at least I find it difficult). As an alternative to losing the information that valued data may provide, the input data could be dichotomized ([Transform>Dichotomize](#)) at various levels of strength. The structural holes measures may be computed for either directed or undirected data -- and the interpretation, of course, depends on which is used. Here, we've used the directed binary data. Three output arrays are produced, and can be saved as separate files (or not, as the output reports all three).

The results are shown in figure 9.8, and need a bit of explanation.

Figure 9.8. Structural holes results for the Knoke information exchange network

Dyadic redundancy										
	1	2	3	4	5	6	7	8	9	10
	COUN	COMM	EDUC	INDU	MAYR	WRO	NEWS	UWAY	WELF	WEST
1	0.00	0.72	0.00	0.61	0.78	0.00	0.72	0.61	0.56	0.39
2	0.43	0.00	0.33	0.47	0.87	0.00	0.57	0.40	0.33	0.40
3	0.00	0.50	0.00	0.50	0.60	0.05	0.70	0.00	0.00	0.35
4	0.61	0.78	0.56	0.00	0.78	0.00	0.61	0.61	0.00	0.00
5	0.44	0.81	0.38	0.44	0.00	0.00	0.56	0.38	0.31	0.31
6	0.00	0.00	0.13	0.00	0.00	0.00	0.38	0.00	0.13	0.00
7	0.54	0.71	0.58	0.46	0.75	0.13	0.00	0.50	0.46	0.38
8	0.69	0.75	0.00	0.69	0.75	0.00	0.75	0.00	0.63	0.00
9	0.63	0.63	0.00	0.00	0.63	0.06	0.69	0.63	0.00	0.00
10	0.50	0.86	0.50	0.00	0.71	0.00	0.64	0.00	0.00	0.00

Dyadic Constraint										
	1	2	3	4	5	6	7	8	9	10
	COUN	COMM	EDUC	INDU	MAYR	WRO	NEWS	UWAY	WELF	WEST
1	0.00	0.13	0.00	0.04	0.15	0.00	0.06	0.04	0.04	0.03
2	0.05	0.00	0.04	0.05	0.11	0.00	0.06	0.04	0.04	0.02
3	0.00	0.09	0.00	0.03	0.10	0.04	0.06	0.00	0.00	0.06
4	0.04	0.13	0.03	0.00	0.13	0.00	0.10	0.04	0.00	0.00
5	0.05	0.09	0.04	0.04	0.00	0.00	0.06	0.04	0.03	0.03
6	0.00	0.00	0.27	0.00	0.00	0.00	0.11	0.00	0.07	0.00
7	0.03	0.10	0.04	0.06	0.11	0.01	0.00	0.03	0.03	0.02
8	0.05	0.15	0.00	0.05	0.15	0.00	0.06	0.00	0.05	0.00
9	0.05	0.13	0.00	0.00	0.13	0.02	0.06	0.05	0.00	0.00
10	0.04	0.08	0.12	0.00	0.17	0.00	0.06	0.00	0.00	0.00

Structural Hole Measures				
	1	2	3	4
	EffSize	Efficie	Constra	Hierarc
1	2.611	0.373	0.481	0.103
2	4.200	0.525	0.401	0.052
3	3.300	0.550	0.386	0.044
4	2.056	0.343	0.479	0.082
5	4.375	0.547	0.387	0.032
6	2.375	0.792	0.454	0.139
7	4.500	0.500	0.424	0.097
8	1.750	0.292	0.514	0.079
9	2.750	0.458	0.436	0.101
10	1.786	0.357	0.486	0.072

Dyadic redundancy means that ego's tie to alter is "redundant." If A is tied to both B and C, and B is tied to C (as in figure 9.5) A's tie to B is redundant, because A can influence B by way of C. The dyadic redundancy measure calculates, for each actor in ego's neighborhood, how many of the other actors in the neighborhood are also tied to the other. The larger the proportion of others in the neighborhood who are tied to a given "alter," the more "redundant" is ego's direct tie. In the example, we see that actor 1's (COUN) tie to actor 2 (COMM) is largely redundant, as 72% of ego's other neighbors also have ties with COMM. Actors that display high dyadic redundancy are actors who are embedded in local neighborhoods where there are few structural holes.

Dyadic constraint is an measure that indexes the extent to which the relationship between ego and each of the alters in ego's neighborhood "constrains" ego. A full description is given in Burt's 1992 monograph, and the construction of the measure is somewhat complex. At the core though, A is constrained by its relationship with B to the extent that A does not have many alternatives (has few other ties except that to B), and A's other alternatives are also tied to B. If A has few alternatives to exchanging with B, and if those alternative exchange partners are also tied to B, then B is likely to constrain A's behavior. In our example constraint measures are not very large, as most actors have several ties. COMM and MAYR are, however, exerting constraint over a number of others, and

are not very constrained by them. This situation arises because COMM and MAYR have considerable numbers of ties, and many of the actors to whom they are tied do not have many independent sources of information.

Effective size of the network (EffSize) is the number of alters that ego has, minus the average number of ties that each alter has to other alters. Suppose that A has ties to three other actors. Suppose that none of these three has ties to any of the others. The effective size of ego's network is three. Alternatively, suppose that A has ties to three others, and that all of the others are tied to one another. A's network size is three, but the ties are "redundant" because A can reach all three neighbors by reaching any one of them. The average degree of the others in this case is 2 (each alter is tied to two other alters). So, the effective size of the network is its actual size (3), reduced by its redundancy (2), to yield an efficient size of 1.

Efficiency (Efficie) norms the effective size of ego's network by its actual size. That is, what proportion of ego's ties to its neighborhood are "non-redundant." The effective size of ego's network may tell us something about ego's total impact; efficiency tells us how much impact ego is getting for each unit invested in using ties. An actor can be effective without being efficient; and an actor can be efficient without being effective.

Constraint (Constra) is a summary measure that taps the extent to which ego's connections are to others who are connected to one another. If ego's potential trading partners all have one another as potential trading partners, ego is highly constrained. If ego's partners do not have other alternatives in the neighborhood, they cannot constrain ego's behavior. The logic is pretty simple, but the measure itself is not. It would be good to take a look at Burt's 1992 Structural Holes. The idea of constraint is an important one because it points out that actors who have many ties to others may actually lose freedom of action rather than gain it -- depending on the relationships among the other actors.

Hierarchy (Hierarc) is another quite complex measure that describes the nature of the constraint on ego. If the total constraint on ego is concentrated in a single other actor, the hierarchy measure will have a higher value. If the constraint results more equally from multiple actors in ego's neighborhood, hierarchy will be less. The hierarchy measure, in itself, does not assess the degree of constraint. But, among whatever constraint there is on ego, it measures the important property of dependency -- inequality in the distribution of constraints on ego across the alters in its neighborhood.

[table of contents](#)

Brokerage

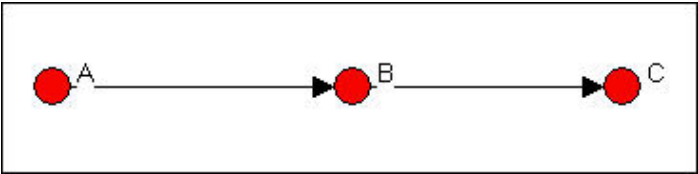
Burt's approach to understanding how the way that an actor is embedded in its neighborhood is very useful in understanding power, influence, and dependency effects. We'll examine some similar ideas in the chapter on centrality. Burt's underlying approach is that of the rational individual actor who may be attempting to maximize profit or advantage by modifying the way in which they are embedded. The perspective is decidedly "neo-classical."

Fernandez and Gould also examined the ways in which actor's embedding might constrain their behavior. These authors though, took a quite different approach; they focus on the roles that ego plays in connecting groups. That is, Fernandez and Gould's "brokerage" notions examine ego's relations with its neighborhood from the perspective of ego acting as an agent in relations among groups (though, as a practical matter, the groups in brokerage analysis can be individuals).

To examine the brokerage roles played by a given actor, we find every instance where that actor lies on the directed path between two others. So, each actor may have many opportunities to act as a "broker." For each one of the instances where ego is a "broker," we examine which *kinds* of actors are involved. That is, what are the group memberships of each of the three actors? There are five possible combinations.

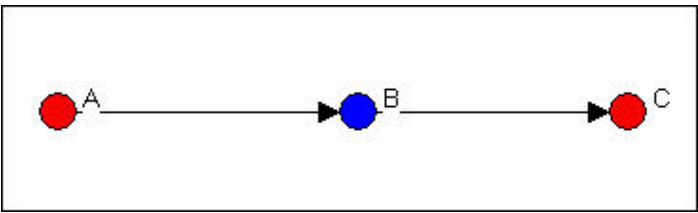
In figure 9.9, the ego who is "brokering" (node B), and both the source and destination nodes (A and C) are all members of the same group. In this case, B is acting as a "coordinator" of actors within the same group as itself.

Figure 9.9. Ego B as "coordinator"



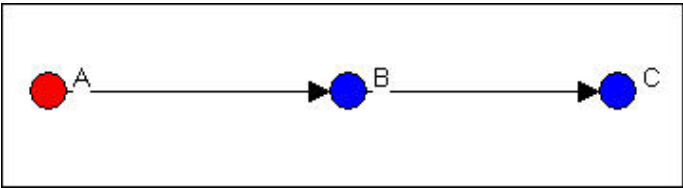
In figure 9.10, ego B is brokering a relation between two members of the same group, but is not itself a member of that group. This is called a "consulting" brokerage role.

Figure 9.10. Ego B as "consultant"



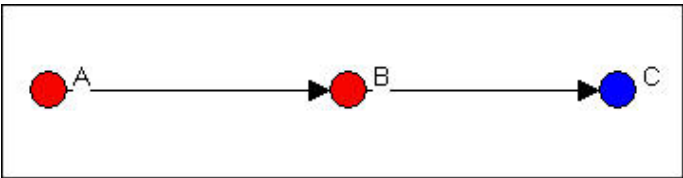
In figure 9.11, ego B is acting as a gatekeeper. B is a member of a group who is at its boundary, and controls access of outsiders (A) to the group.

Figure 9.11. Ego B as "gatekeeper"



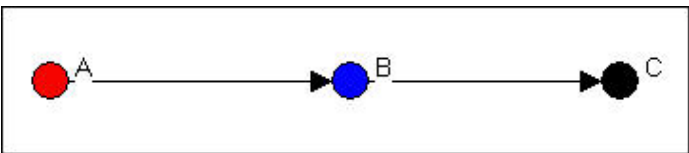
In figure 9.12, ego B is in the same group as A, and acts as the contact point or representative of the red group to the blue.

Figure 9.12. Ego B as "representative"



Lastly, in figure 9.13, ego B is brokering a relation between two groups, and is not part of either. This relation is called acting as a "liaison."

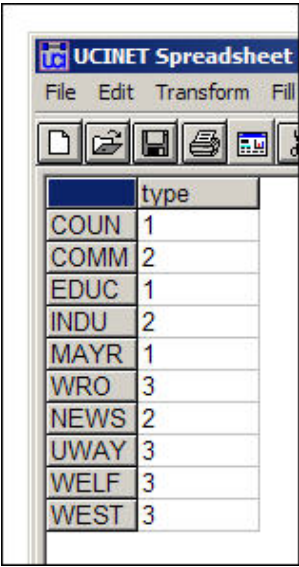
Figure 9.13. Ego B as "liaison"



To examine brokerage, you need to create an attribute file that identifies which actor is part of which group. You can select one of the attributes from a user-created attribute file, or use output files from other UCINET routines that store descriptors of nodes as attributes. As an example, we've taken the Knoke information exchange network, and classified each of the organizations as either a general government organization (coded 1), a private

non-welfare organization (coded 2), or an organizational specialist (coded 3). Figure 9.14 shows the attribute (or partition) as we created it using the UCINET spreadsheet editor.

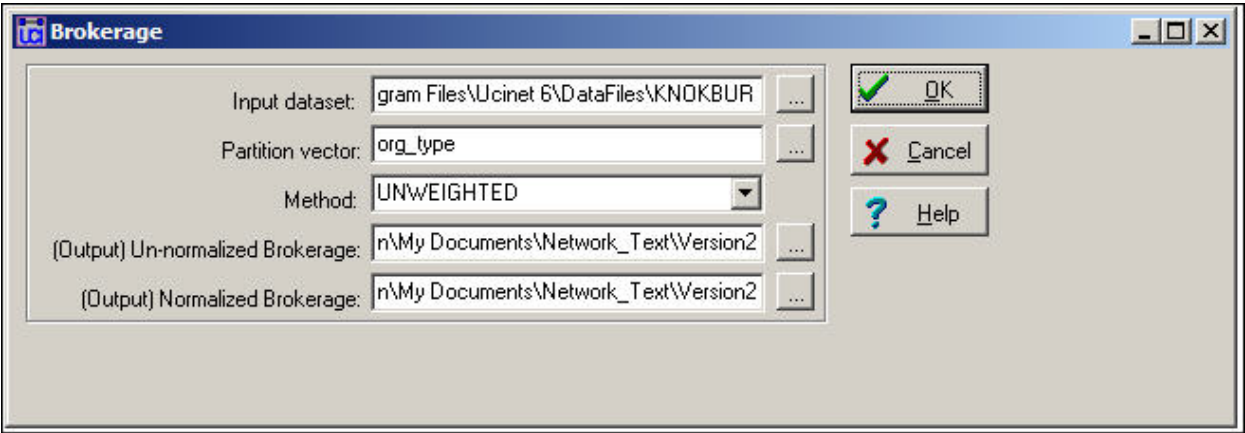
Figure 9.14. Partition vector for Knoke information exchange



The screenshot shows the UCINET Spreadsheet application window. It contains a table with two columns: 'type' and a list of organizational codes. The codes and their corresponding type values are as follows:

	type
COUN	1
COMM	2
EDUC	1
INDU	2
MAYR	1
WRO	3
NEWS	2
UWAY	3
WELF	3
WEST	3

Using the network data set and the attribute vector we just created, we can run *Network>Ego Networks>Brokerage*, as shown in figure 9.15.



The screenshot shows the 'Brokerage' dialog box in UCINET. The settings are as follows:

- Input dataset: gram Files\Ucinet 6\DataFiles\KNOKBUR
- Partition vector: org_type
- Method: UNWEIGHTED
- (Output) Un-normalized Brokerage: n\My Documents\Network_Text\Version2
- (Output) Normalized Brokerage: n\My Documents\Network_Text\Version2

Buttons on the right include OK, Cancel, and Help.

The option "unweighted" needs a little explanation. Suppose that actor B was brokering a relation between actors A and C, and was acting as a "liaison." In the unweighted approach, this would count as one such relation for actor B. But, suppose that there was some other actor D who also was acting as a liaison between A and C. In the "weighted" approach, both B and D would get 1/2 of the credit for this role; in the unweighted approach, both B and D would get full credit. Generally, if we are interested in ego's relations, the unweighted approach would be used. If we were more interested in group relations, a weighted approach might be a better choice.

The output produced by *Network>Ego Networks>Brokerage* is quite extensive. We'll break it up into a few parts and discuss them separately. The first piece of the output (figure 9.16) is a census of the number of times that each actor serves in each of the five roles.

Figure 9.16. Unnormalized brokerage scores for Knoke information network

	1	2	3	4	5	6
	Coordinat	Gatekeepe	Represent	Consultan	Liaison	Total
1	0	0	0	1	1	2
3	0	1	1	2	5	9
5	2	6	5	5	9	27
2	0	3	7	5	6	21
4	0	0	1	1	0	2
7	0	5	0	0	1	6
6	0	1	0	0	0	1
8	0	0	0	0	0	0
9	0	0	2	0	0	2
10	0	0	0	1	0	1

The actors have been grouped together into "partitions" for presentation; actors 1, 3, and 5, for example, form the first type of organization. Each row counts the raw number of times that each actor plays each of the five roles in the whole graph. Two actors (5 and 2) are the main sources of inter-connection among the three organizational populations. Organizations in the third population (6, 8, 9, 10), the welfare specialists, have overall low rates of brokerage. Organizations in the first population (1, 3, 5), the government organizations seem to be more heavily involved in liaison than other roles. Organizations in the second population (2, 4, 7), non-governmental generalists play more diverse roles. Overall, there is very little coordination within each of the populations.

We might also be interested in how frequently each actor is involved in relations among and within each of the groups. Figure 9.17 shows these results for the first two nodes.

Figure 9.17. Group-to-group brokerage map

Node 1 (group 1)									
		1	2	3					
		1	2	3					
		-	-	-					
1	1	0	0	0					
2	2	0	0	1					
3	3	0	0	1					
Node 2 (group 2)									
		1	2	3					
		1	2	3					
		-	-	-					
1	1	2	1	3					
2	2	3	0	4					
3	3	3	2	3					

We see that actor 1 (who is in group 1) plays no role in connections from group 1 to itself or the other groups (i.e. the zero entries in the first row of the matrix). Actor 1 does, however, act as a "liaison" in making a connection from group 2 to group 3. Actor 1 also acts as a "consultant" in connecting a member of group 3 to another member of group 3. The very active actor 2 does not broker relations within group 2, but is heavily involved in ties in both directions of all three groups to one another, and relations among members of groups 1 and 3.

These two descriptive maps can be quite useful in characterizing the "role" that each ego is playing in the relations among groups by way of their inclusion in its local neighborhood. These roles may help us to understand how each ego may have opportunities and constraints in access to the resources of the social capital of groups, as well as individuals. The overall maps also inform us about the degree and form of cohesion within and between the groups.

There may be some danger of "over interpreting" the information about individuals brokerage roles as representing meaningful acts of "agency." In any population in which there are connections, partitioning will produce brokerage - even if the partitions are not meaningful, or even completely random. Can we have any confidence that the

patterns we are seeing in real data are actually different from a random result?

In Figure 9.18, we see the number of relations of each type that would be expected by pure random processes. We ask: what if actors were assigned to groups as we specify, and each actor has the same number of ties to other actors that we actually observe; but, the ties are distributed at random across the available actors? What if the pattern of roles was generated entirely by the number of groups of various sizes, rather than representing efforts by the actors to deliberately construct their neighborhoods to deal with the constraints and opportunities of group relations?

Figure 9.18. Expected values under random assignment

Expected Values (given number of groups and sizes of each group)						
	1	2	3	4	5	6
	Coordinat	Gatekeepe	Represent	Consultan	Liaison	Total
1	0.100	0.433	0.433	0.433	0.600	2.000
3	0.450	1.950	1.950	1.950	2.700	9.000
5	1.350	5.850	5.850	5.850	8.100	27.000
2	1.050	4.550	4.550	4.550	6.300	21.000
4	0.100	0.433	0.433	0.433	0.600	2.000
7	0.300	1.300	1.300	1.300	1.800	6.000
6	0.050	0.217	0.217	0.217	0.300	1.000
8	0	0	0	0	0	0
9	0.100	0.433	0.433	0.433	0.600	2.000
10	0.050	0.217	0.217	0.217	0.300	1.000

If we examine the actual brokerage relative to this random expectation, we can get a better sense of which parts of which actors roles are "significant." That is, occur much more frequently than we would expect in a world characterized by groups, but random relations among them.

Figure 9.19. Normalized brokerage scores

Relative Brokerage (raw scores divided by expected values given group sizes)						
	1	2	3	4	5	6
	Coordinat	Gatekeepe	Represent	Consultan	Liaison	Total
1	0	0	0	2.308	1.667	1.000
3	0	0.513	0.513	1.026	1.852	1.000
5	1.481	1.026	0.855	0.855	1.111	1.000
2	0	0.659	1.538	1.099	0.952	1.000
4	0	0	2.308	2.308	0	1.000
7	0	3.846	0	0	0.556	1.000
6	0	4.615	0	0	0	1.000
8	0	0	0	0	0	0
9	0	0	4.615	0	0	1.000
10	0	0	0	4.615	0	1.000

The normalized brokerage scores in this example need to be treated with a little caution. As with most "statistical" approaches, larger samples (more actors) produce more stable and meaningful results. Since our network does not contain large numbers of relations, and does not have high density, there are many cases where the expected number of relations is small, and finding no such relations empirically is not surprising. Both actor 2 and actor 5, who do broker many relations, do not have profiles that differ greatly from what we would expect by chance. The lack of large deviations from expected values suggests that we might want to have a good bit of caution in interpreting our seemingly interesting descriptive data as being highly "significant."

[table of contents](#)

Summary

In this chapter we've taken another look at the notion of embedding; this time, our focus has been on the individual actor, rather than the network as a whole.

The fundamental idea here is that the ways in which individuals are attached to macro-structures is often by way of their local connections. It is the local connections that most directly constrain actors, and provide them with access to opportunities. Examining the ego-networks of individuals can provide insight into why one individual's perceptions, identity, and behavior differ from another's. Looking at the demography of ego networks in a whole population can tell us a good bit about its differentiation and cohesion - from a micro point of view.

In the next several chapters we will examine additional concepts and algorithms that have been developed in social network analysis to describe important dimensions of the ways in which individuals and structures interact. We'll start with one of the most important, but also most troublesome, concepts: power.

[table of contents](#)

[table of contents of the book](#)