Chapter 4
Descriptive Analysis of Network Graph Characteristics

Given a network graph representation of a system – possibly constructed from an appropriate set of measurements, in the manner of the previous chapter – it is common to explore the characteristics and structural properties of the network. Tasks range from the calculation of simple metrics summarizing topological structure, both local and global, to the unsupervised extraction of complex relational patterns. In this chapter we present a core set of tools and techniques for such purposes.

4.1 Introduction

In the study of a given complex system, questions of interest can often be re-phrased in a useful manner as questions regarding some aspect of the structure or characteristics of a corresponding network graph. For example, various types of basic social dynamics can be represented by triplets of vertices with a particular pattern of ties among them (i.e., triads); questions involving the movement of information or commodities usually can be posed in terms of paths on the network graph and flows along those paths; certain notions of the 'importance' of individual system elements may be captured by measures of how 'central' the corresponding vertex is in the network; and the search for 'communities' and analogous types of unspecified 'groups' within a system frequently may be addressed as a graph partitioning problem.

The structural analysis of network graphs has traditionally been treated primarily as a descriptive task, as opposed to an inferential task, and the tools commonly used for such purposes derive largely from areas outside of 'mainstream' statistics. For example, an overwhelming proportion of these tools are naturally graph-theoretic in nature, and thus have their origins in mathematics and computer science.[1] Similarly, the field of social network analysis has been another key source, contributing tools usually aimed – at least originally – at capturing basic aspects of social structure and

[1] The concepts and terminology reviewed in Chapter 2.1 will be particularly relevant here in the present chapter.

dynamics. More recently, the field of physics has also been an important contributor, with the proposed tools often motivated by analogues in statistical mechanics.

For convenience of exposition, we have organized the various concepts and tools for structural analysis to be covered in this chapter as primarily falling into two broad categories: (i) characterization of individual vertices and edges, and (ii) characterization of network cohesion (i.e., involving more than just individual vertices and edges). Note that we have expressly *not* used the related terms 'local' and 'global' in our categorization, as some measures of vertex-level characteristics in fact summarize information from across the graph, and conversely there can be cohesion among vertices only in some small portion of the graph.

By way of organization, this chapter is laid out as follows. Section 4.2 covers vertex and edge characteristics, while Section 4.3 covers the characterization of network cohesion. Then, in Section 4.4, we illustrate the combined usage of the various tools and techniques described, through a case study in the context of networks of cortical-level probes in the brain and epileptic seizures. Lastly, the analysis of structure in dynamic networks is touched upon in Section 4.5. Some additional topics, as well as references for reading, are described in Section 4.6.

4.2 Vertex and Edge Characteristics

As the fundamental elements of network graphs are their vertices and edges, there are a number of network characterizations centered upon these. We discuss several such characterizations here in this section. Our presentation is broken down according to (i) those characterizations based upon vertex degrees, and (ii) those seeking to capture some more general notion of the 'importance' of a vertex – typically referred to as vertex centrality measures. We also discuss how measures of vertex centrality often extend in a straightforward manner to describe the importance of edges.

4.2.1 Degree

Recall that the degree d_v of a vertex v, in a network graph $G = (V,E)$, counts the number of edges in E incident upon v. That is, it provides a basic quantification of the extent to which v is connected to other vertices within the graph. When the vertex degrees are considered in aggregate, through the degree sequence $\{d_1,\ldots,d_{N_v}\}$, various useful measures of the nature of the overall connectivity in the graph can be defined. We discuss a few such measures here. Note that in the case of directed graphs, such measures are typically applied separately to the sequences $\{d_v^{in}\}_{v\in V}$ and $\{d_v^{out}\}_{v\in V}$ of in- and out-degrees.

4.2.1.1 Degree Distributions

Given a network graph G, define f_d to be the fraction of vertices $v \in V$ with degree $d_v = d$. The collection $\{f_d\}_{d \geq 0}$ is called the *degree distribution* of G, and is simply the histogram formed from the degree sequence, with bins of size one, centered on the non-negative integers. For example, in the protein-protein interaction network of Figure 3.2, corresponding to *Drosophila*'s circadian clock mechanism, the degree sequence $\{1,3,4,4,3,3\}$, associated with the vertices *Sgg* through *Cyc*, moving clockwise, yields the degree distribution $\{f_1 = 1/6,\, f_3 = 1/2,\, f_4 = 1/3\}$.

For directed graphs, degree distributions may be defined analogously for in- and out-degrees. For example, in the gene regulatory network also shown in Figure 3.2, the in-degree distribution is $\{f_2^{in} = 2/3,\, f_3^{in} = 1/3\}$, while the out-degree distribution is $\{f_0^{out} = 1/3,\, f_2^{out} = 1/3,\, f_4^{out} = 1/3\}$.

The degree distribution provides a natural summary of the connectivity in the graph. The examples above are, of course, fairly trivial. In practice, degree distributions are arguably more interesting as descriptors for large graphs.

Example 4.1 (Degree Structure for Internet and Protein Interaction Networks). Figure 4.1 shows plots of the degree distributions for two large network graphs. The first graph corresponds to the router-level Internet representation derived from the CAIDA data described in Section 3.5.2. Recall that this graph derived from the union of measured subgraphs, resulting from probes sent from roughly 20 measurement centers to approximately $800,000$ Internet destinations, resulting in $192,244$ discovered vertices and $609,066$ discovered edges.

The second graph corresponds to a network of measured protein interactions in the organism *Saccharomyces cerevisiae* – better known as brewers' or bakers' yeast. The data were culled from the January 2007 version of the BioGRID database,[2] a general repository for biological interaction datasets, and consist of an aggregation of published experimental results from various labs across the world on the affinity for protein pairs to interact with each other. This network graph, made up of $5,151$ vertices (proteins) and $31,201$ edges (interactions), is similar to the protein-interaction network for *Drosophila* shown in Figure 3.2. However, it differs not only in magnitude but also, importantly, in the nature of the declared interactions. Specifically, in the yeast database, the interactions were determined largely by the strength of evidence provided by various high-throughput experimental techniques, rather than cumulative biological judgment. In addition, the proteins and their interactions are not specific to any one biological function, whereas in Figure 3.2, all proteins and interactions are present precisely due to their role in the *Drosophila* circadian clock mechanism.

One aspect of these two networks that is immediately apparent is the broad range over which the degrees in the graphs vary. The vertices in the protein network range in degree from 1 to 301, and those in the Internet graph, from 1 to $1,071$. In addition, it turns out that both degree distributions are highly skewed to the right. The average vertex degree $\bar{d}$ is roughly 6.3 and 12 for the Internet and protein interaction data,

[2] See *http://www.thebiogrid.org/* and Stark et al. [367].

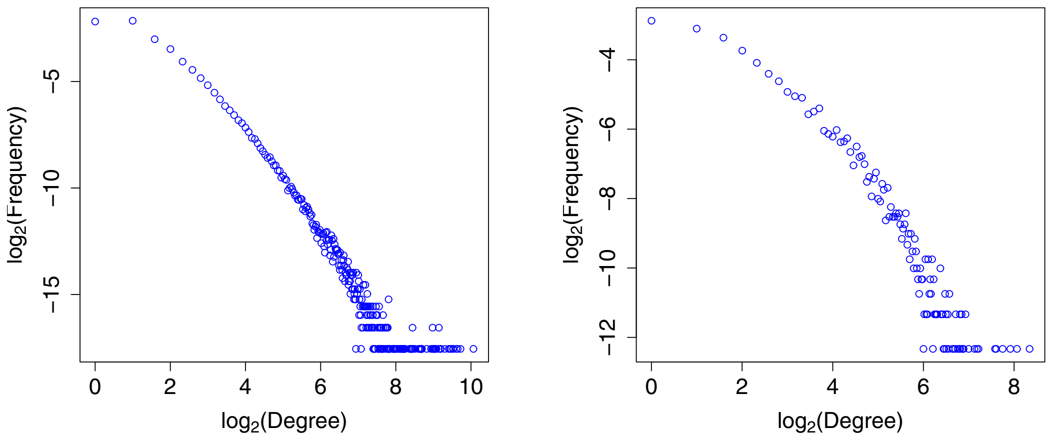

Fig. 4.1 Degree distributions. Left: the router-level Internet network graph described in Section 3.5.2. Right: the network of measured interactions among proteins in *S. cerevisiae* (yeast), as of January 2007. In each plot, both x- and y-axes are in base-2 logarithmic scale.

respectively, while the corresponding medians are only 3 and 6. The averages are actually much closer to the third quartiles, found at 6 and 15, respectively.

In fact, each of these distributions has been plotted on a logarithmic scale on both axes (i.e., on a log-log scale), and as a result, we see that the skewness in these distributions is of a particular nature. Specifically, while the majority of vertices are of very low degree, a nevertheless nontrivial number of vertices are of much higher degree – two to three orders of magnitude higher, in some cases. The roughly linear decay of the points in each plot, over at least some portion of the range of each distribution, suggests the presence of a power-law component to these distributions. That is, it suggests that something like

$$f_d \propto d^{-\alpha} \tag{4.1}$$

holds approximately true in part. □

In just the past 10 or 15 years, it has been found that approximate power-law degree distributions appear to be ubiquitous in networks across many areas of the sciences. This discovery was originally quite unexpected, as such structure is in direct contrast to that of networks typically studied throughout much of the 20th century, such as networks of circuits or traditional random graphs.[3] In these latter cases, vertex degree is of a fairly similar order of magnitude across the graph – homogeneous, rather than heterogeneous. The corresponding degree distributions are hence quite concentrated, rather than diffuse, and typically decay exponentially fast in d, rather than like a power-law. Note that the emergence of power-law-like behavior in network graphs is not simply a function of increasing network size. For example, the

[3] We will encounter random graphs of this type in Chapter 6.

power-grid of the western United States, forming a network of roughly $5,000$ generators, transformers and substations, connected by just over $13,000$ transmission lines, has a degree distribution with exponential decay (e.g., see Newman [296, Fig. 6]).

In any event, given an observed degree distribution, whether homogeneous or heterogeneous, it is tempting – and indeed natural – to want to summarize that distribution. Reporting basic summary statistics, in the form of moments and quantiles, as in Example 4.1, is common. It is also common in some fields to report the fit of some simple, parametric family of distributions to the observed degree distribution. This task, however, can be far from trivial when heterogeneous degree distributions are involved.

Example 4.2 (Fitting Power-Laws to the Internet and Protein-Interaction Networks). Consider the degree distributions for the Internet and protein-interaction networks shown in Figure 4.1, and recall that their shape suggested the possibility of a power-law distribution, like that in (4.1), over at least some portion of their support. Ignoring, for the moment, issues regarding the support of the power-law component, the exponent α is the key parameter that needs to be extracted from the data.

There are a number of ways to proceed in this task, although some are more advisable than others. Most simply, we could note that (4.1) implies the approximate relation

$$\log f_d \sim C - \alpha \log d \quad , \tag{4.2}$$

where C is an arbitrary constant. Hence, in principle, a sensible estimate of α might be obtained by fitting a line to the plot of $\log f_d$ versus $\log d$, using a method like least squares regression. In practice, however, this method is not advisable, due to the disproportionate level of 'noise' in the data at the high degrees, where the levels of counts are the lowest.

A common solution to the noise problem is to use cumulative frequencies, rather than the raw frequencies themselves, a process which effectively smooths the noise. For example, since under model (4.1) the tail probabilities take the form

$$\bar{F}(d) = 1 - F(d) \sim d^{-(\alpha-1)} \quad , \tag{4.3}$$

we can then again consider using a regression-based approach to estimate α. Alternatively, in the statistical physics literature there is the related practice of using relative frequencies calculated on intervals of logarithmically increasing size (i.e., so-called logarithmic binning), followed by a linear regression fit. Details may be found in Pastor-Satorras and Vespignani [309, App. 3], for example.

All of these regression-based approaches, although common, have been criticized for their somewhat *ad hoc* nature and the biases that can be introduced. See the numerical study by Goldstein, Morris, and Yen [179], for example. A more rigorous alternative, and one widely used in many fields that routinely encounter power-laws, are estimators of the form

$$\hat{\alpha}_k = 1 + \hat{\gamma}_k^{-1} \quad , \quad \text{with} \quad \hat{\gamma}_k = \frac{1}{k} \sum_{i=0}^{k-1} \log \frac{d_{(N_v - i)}}{d_{(N_v - k)}} \quad , \tag{4.4}$$

where $d_{(1)} \leq \cdots \leq d_{(N_v)}$ are the sorted vertex degrees and k is a value chosen by the user.[4] Typically, instead of a single choice of k, the estimate (4.4) is computed over a range of choices for k, and a plot of $\hat{\alpha}_k$ versus k – called a *Hill plot* – is examined with an eye towards finding areas where the plot 'settles down' to some stable values of $\hat{\alpha}$, away from extremes where k is too small or too large. Drees, de Haan, and Resnick [126] provide a discussion of these and similar plots.

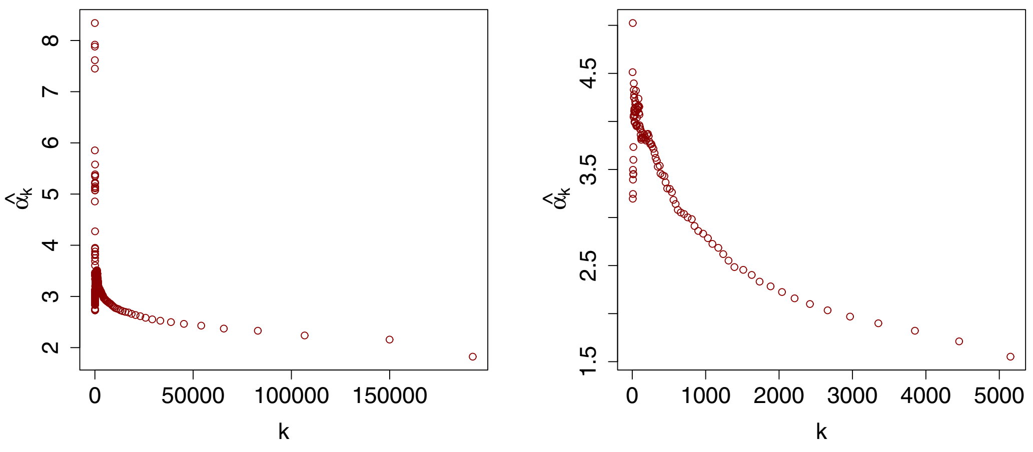

Fig. 4.2 Hill plots of maximum likelihood estimates $\hat{\alpha}_k$, as a function of k (i.e., the number of order statistics used), for the Internet (left) and protein interaction (right) datasets.

Hill plots are shown for our two networks in Figure 4.2. The plot for the Internet network shows a broad stretch of estimates with fairly stable behavior, having just a slight decay from 3 to 2 over many orders of magnitude in k, where the limiting behavior in this range (as $k \to 0$) arguably tends towards a value a bit below 3.0. This sort of appearance, and interpretation, is fairly typical of Hill plots where a power-law model is reasonably credible. In contrast, the plot for the protein interaction network looks rather different, with estimates decaying much more sharply than in the Internet network, and over a decidely smaller range of values k. Looking back at Figure 4.1, we note that the degree distribution for the protein interaction network arguably has a more concave nature to it than that for the Internet network. This concavity produces the corresponding convexity in the Hill plot, which

[4] The estimator $\hat{\gamma}$ is called the *Hill estimator*. When the model (4.1) holds exactly, it is the maximum likelihood estimate of $\gamma = (\alpha - 1)^{-1}$; more generally, it is an approximate maximum likelihood estimate if $1 - F(d) \sim d^{-(\alpha-1)} L(d)$, for some slowly varying function $L(\cdot)$, given appropriate choice of k. Consistency of this estimator has been proved under a variety of conditions. See Drees, de Haan, and Resnick [126].

strongly suggests that a simple power-law-like model is inappropriate in the case of the protein network. □

Exponents summarizing power-law-like behavior in network degree distributions should be reported with caution, and reported exponents should be interpreted with the proverbial grain of salt.[5] Note that to even realistically entertain discussion of exponents of this type is only really appropriate for large networks in which the degrees range over a good few orders of magnitude, the number of which is intimately tied to the value α itself. In many settings, it may not even be feasible that vertices possess a degree beyond some modest value d_{max}. For example, while we observed earlier that the largest degree in our Internet network graph was $1,071$, at the time that the underlying dataset was collected most routers generally could establish no more than 64 links with other routers, or perhaps 128 in special cases. 'Routers' in our graph with larger vertex degree than this are actually composites of many such routing devices. Hence, the interpretation of the results above require some care. Similarly, in the context of social networks, there is often a ceiling on the number of relationships of a given type that an individual can maintain. Most social relationships of substantive interest require personal resources of some nature, and so too many relationships become costly.

In the face of such concerns, a seemingly obvious solution is to consider more complicated models, such as mixtures of power-laws or distributions with power-law behavior and exponential truncation (i.e., $f_d \propto d^{-\alpha}\exp(-d/d^*)$, for some $d^* \gg 0$). Unfortunately, fitting such models and, more broadly, discriminating between various hypothesized models of these types, is usually a delicate task at best. Of course, the challenge here is not restricted to the realm of networks alone, but rather is due to working with power-law-like distributions (i.e., distributions possessing long, heavy tails). See Hernández-Campos, Marron, Samorodnitsky, and Smith [197], for example, for discussion of some of the underlying issues and some proposed solutions, in the context of heavy-tailed Internet traffic data, as well as the references therein. Also see Mitzenmacher [280] for an extensive history of power-law-like distributions in general and discussion of some of the various subtleties and challenges in this area.

On a final note, we point out that, beyond the issue of appropriately summarizing power-law-like degree distributions, it is a topic of intense interest to understand the mechanisms by which such distributions emerge, both from theoretical and practical perspectives. We will return to this topic in Chapter 6. Additionally, there has also been much interest in understanding the impact of this structure on such things as communication, search, and the spread of epidemics in networks, topics we will pick up in Chapter 8.

[5] We note too that, given such concerns about the estimates of α themselves, the tendency in the literature to also report standard errors for these estimates is somewhat gratuitous.

4.2.1.2 Degree Correlation

The degree distribution is useful as a composite summary of how degree varies across vertices in the graph, but it does not provide any information on precisely which vertices are connected to which others. To capture information of this sort, it is helpful to establish summaries that describe the patterns of association among vertices of given degrees. In fact, such summaries can be quite important, as two graphs may have identical degree sequences and yet otherwise differ noticeably in the way their vertices are paired. See Doyle et al. [125], for example, for a study of this phenomenon and its implications in the context of the topology of the Internet.

A natural starting point is to define a two-dimensional analogue of the degree distribution, capturing the relative frequency with which the two vertices at the end of an arbitrarily selected edge in the graph have a given pair of degrees. For undirected graphs, some care is needed in defining this distribution, in light of the lack of ordering among vertices that characterize an edge. One approach is to first sort the vertices in all edges $e \in E$ so as to have the form $e = (v_1, v_2)$, where $d(v_1) \le d(v_2)$. Then, for each pair $d_1 < d_2$, we assign half of the relative frequency of the sorted edges with degrees (d_1, d_2) to f_{d_1,d_2} and the other half to f_{d_2,d_1}. For $d_1 = d_2$, we let f_{d_1,d_2} simply be the relative frequency of edges with both vertices of this degree. The resulting distribution, say $\{f_{d,d'}\}$, is symmetric and corresponds to selecting an edge at random from E and reporting it with its vertices in increasing degree or decreasing degree with equal chance.[6]

Example 4.3 (Joint Degree Distributions for the Internet and Protein Interaction Networks). Figure 4.3 shows image representations of the joint degree distributions for the router-level Internet data and the protein interaction data. In both cases we find the joint distribution to be concentrated primarily where pairs (d, d') are both low, as would be expected from our study of the marginal degree distributions. However, interestingly, we can see there is also a noticeable tendency for the vertices of largest degree to be connected to low-degree vertices, rather than other high-degree vertices. The effect appears to be somewhat more pronounced in the protein interaction data. Maslov and Sneppen [270] and Maslov, Sneppen, and Zaliznyak [271] originally made observations of this kind in earlier protein and Internet datasets, where they examined a normalized version of the joint degree distribution we have defined here. □

From the joint degree distribution we can define other useful summaries. For example, we might examine the distribution of the degrees of the neighbors of a vertex, conditional on the degree of that vertex. That is, we might examine relative frequencies of the form $f_{d'|d}$ with which edges from a vertex of degree d are connected to another vertex of degree d'. Plots of the means of these conditional distributions,

[6] Our definition corresponds to the natural interpretation of an undirected graph as a symmetric directed graph. For general directed graphs, a separate joint degree distribution can be defined for in- and out-degrees, although it is no longer expected that these be symmetric. Alternatively, we can choose just to work with an undirected version of the graph, with degrees $d_i = d_i^{in} + d_i^{out}$.

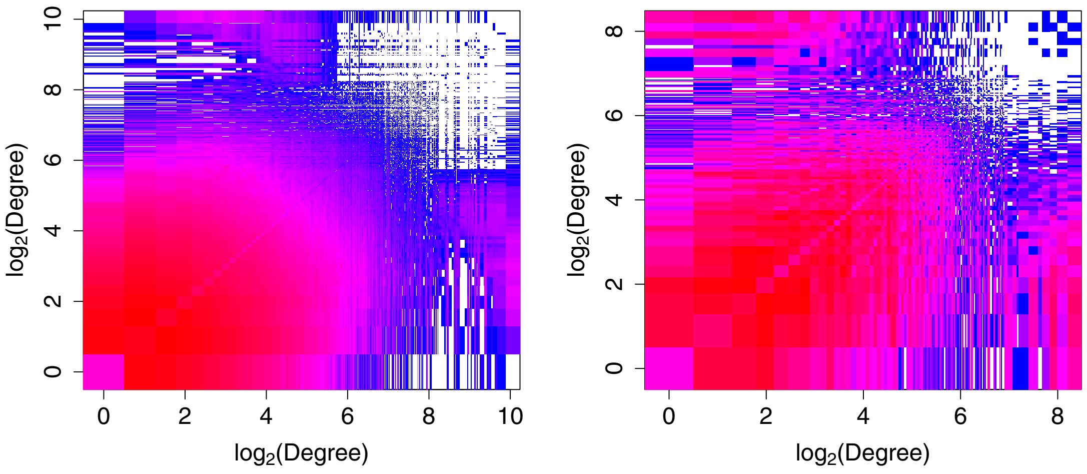

Fig. 4.3 Image representation of the logarithmically transformed joint degree distribution $\{\log_2 f_{d,d'}\}$, for the router-level Internet data (left) and the protein interaction data (right). Colors range from blue (low relative frequency) to red (high relative frequency), with white indicating areas with no data. Note that both x- and y-axes are also on base-2 logarithmic scales.

$$\bar{d}(d) = \sum_{d'} d' f_{d'|d} \quad , \tag{4.5}$$

as a function of d, have been found useful in concisely summarizing the type of association between high- and low-degree vertices remarked upon in Example 4.3. In particular, a negative trend has been observed in $\bar{d}(d)$ as d increases (e.g., Maslov and colleagues [270, 271], Pastor-Satorras, Vázquez, and Vespignani [310], Vázquez, Pastor-Satorras, and Vespignani [387]).

A single number summary of the association between degrees of vertices on an arbitrary edge is provided by the correlation, say $\mathrm{corr}(D, D')$, defined by the joint degree distribution $f_{d,d'}$ and its marginals. For the Internet and protein interaction networks, the degree correlation is found to be 0.023 and -0.083, respectively. While both of these are relatively small, the difference in sign seems to reinforce our observation above that the tendency of high-degree vertices to pair with vertices of low degree is somewhat more pronounced in the protein interaction network.

Newman [288] has defined this same degree correlation through the concept of the 'remaining degrees' of vertices (i.e., the number of edges incident to a vertex, excluding the edge by which one arrives at that vertex). And Doyle et al. [125] have defined another related measure, one they call the 'degree likelihood,' which is proportional to the sum, over all edges, of the products $d(v_1)d(v_2)$ of the degrees of the vertices v_1, v_2 defining the edges $e = (v_1, v_2)$.

Of course, we would be remiss not to point out before closing this section that, while all of these various notions of degree correlation – both visual and numerical – can be useful, it will nonetheless typically be important to look beyond the numbers to ask the more interesting question of *why* an observed correlation is as it is. More likely than not, the association among vertex degrees is actually driven by parallel associations among certain latent vertex characteristics. For example, the function or structure of proteins, or the placement and engineering of a router (e.g., in the Internet backbone, say, versus comparative ‘byways’), are likely to be more meaningful characteristics in our examples above than degree. In the event that such characteristics are measured, there are related notions of correlation – more commonly termed ‘assortativity’ in this context – that may be quantified, as we will see in Section 4.3.4.

4.2.2 Centrality

Many questions that might be asked about a vertex in a network graph essentially seek to understand its ‘importance’ in the network. Which actors in a social network seem to hold the ‘reins of power’? How authoritative does a particular page in the World Wide Web seem to be considered? The deletion of which genes in a gene regulatory network is likely to be lethal to the corresponding organism? How critical is a given router in an Internet network to the flow of traffic? Measures of *centrality* are designed to quantify such notions of ‘importance’ and thereby facilitate the answering of such questions.

There are a vast number of different centrality measures that have been proposed over the years. As observed by Freeman [157] in 1979, and evidently still true today, “There is certainly no unanimity on exactly what centrality is or on its conceptual foundations, and there is little agreement on the proper procedure for its measurement.” Attempts to categorize different centrality measures have involved trying to group them according to concepts or dynamics, like ‘distance,’ ‘flow,’ ‘feedback,’ and ‘control,’ as in Freeman [157] and the volume edited by Brandes and Erlebach [54, Ch. 3]; or, alternatively, by the manner in which they are computed, as in Borgatti and Everett [48].

We have already encountered what is arguably the most widely used measure of vertex centrality: vertex degree. Here we will focus our discussion primarily around the most common versions of three other classic types of vertex centrality measures – typically termed closeness, betweenness, and eigenvector centrality, respectively. Afterward we will comment on extensions, including notions of edge centrality.

4.2.2.1 Three Common Vertex Centrality Measures

We will assume that G is an undirected graph; extensions of the measures below to directed graphs are straightforward. One common notion of ‘central’ is that a vertex be ‘close’ to many other vertices. *Closeness centrality* measures attempt to capture this notion. The standard approach, introduced by Sabidussi [336], is to let the centrality vary inversely with a measure of the total distance of a vertex from all others,

$$c_{Cl}(v) = \frac{1}{\sum_{u \in V} \mathsf{dist}(v,u)} \quad , \tag{4.6}$$

where $\mathsf{dist}(v,u)$ is the geodesic distance between the vertices $u,v \in V$. Often, for comparison across graphs and with other centrality measures, this measure is normalized to lie in the interval $[0,1]$, through multiplication by a factor $N_v - 1$.

To calculate the centrality in (4.6), for all $v \in V$, we need the shortest path distances between all pairs of vertices in G, which may be computed in $O(N_v^2 \log N_v + N_v N_e)$ time – essentially $O(N_v^2 \log N_v)$ on sparse graphs – using Dijkstra's algorithm, as discussed in Chapter 2.1.4. Note that implicitly this measure assumes that the graph G is connected, as otherwise all vertices in principle will have centrality $c_{Cl}(v) = 0$, being of infinite distance from at least one other vertex. A conservative solution to the problem of insufficient connectivity would be to compute centrality indices separately for vertices in each connected component. Or, for example, if the graph consisted primarily of a single larger component, and a handful of other much smaller components, it might be reasonable to restrict attention to just the larger component. Another solution is to set a finite upper limit on distances between vertices; a choice of N_v is appealing, as clearly no vertices can be further than $N_v - 1$ from each other within a connected graph.

Another popular class of centralities are based upon the perspective that 'importance' relates to where a vertex is located with respect to the paths in the network graph. If we picture those paths as the routes by which, say, communication of some sort or another takes place, vertices that sit on many paths are likely more critical to the communication process. *Betweenness centrality* measures are aimed at summarizing the extent to which a vertex is located 'between' other pairs of vertices. The most commonly used betweenness centrality, introduced by Freeman [156], is defined as

$$c_B(v) = \sum_{s \neq t \neq v \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)} \quad , \tag{4.7}$$

where $\sigma(s,t|v)$ is the total number of shortest paths between s and t that pass through v, and $\sigma(s,t) = \sum_v \sigma(s,t|v)$. In the event that shortest paths are unique, $c_B(v)$ just counts the number of shortest paths going through v. This centrality measure can be restricted to the unit interval through division by a factor of $(N_v - 1)(N_v - 2)/2$.

Calculation of the collection of all betweenness centralities $c_B(v)$ requires (i) calculating the lengths of shortest paths among all pairs of vertices, and (ii) computing the summation in (4.7) for each vertex. As a result, the overall computational burden is dominated by the latter step, if implemented in a straightforward fashion, as each of the n sums require $O(N_v^2)$ time. The result is therefore an $O(N_v^3)$ algorithm, which can be prohibitive for large network graphs. However, Brandes [51] offers an algorithm for betweenness centrality that runs in as little as $O(N_v N_e)$ time on unweighted graphs – and $O(N_v^2 \log N_v + N_v N_e)$ time on weighted graphs – as well as requiring only $O(N_v + N_e)$ space, as compared to the $O(N_v^2)$ requirement of the straightforward algorithm.[7] These improvements derive from exploiting a clever recursive relation for the partial sums $\sum_{t \in V} \sigma(s,t|v)/\sigma(s,t)$.

A third class of centrality measures are based on notions of 'status' or 'prestige' or 'rank.' That is, they seek to capture the idea that the more central the neighbors of a vertex are, the more central that vertex itself is. These measures are inherently implicit in their definition and typically can be expressed in terms of eigenvector solutions of appropriately defined linear systems of equations. There are many such *eigenvector centrality* measures. For example, Bonacich [47], following work of Katz [220] and others, defined a centrality measure of the form

$$c_{Ei}(v) = \alpha \sum_{\{u,v\} \in E} c_{Ei}(u) \ . \tag{4.8}$$

The vector $\mathbf{c}_{Ei} = (c_{Ei}(1), \ldots, c_{Ei}(N_v))^T$ is the solution to the eigenvalue problem $\mathbf{A}\mathbf{c}_{Ei} = \alpha^{-1}\mathbf{c}_{Ei}$, where $\mathbf{A}$ is the adjacency matrix for the network graph G. Bonacich [47] argues that an optimal choice of α^{-1} is the largest eigenvalue of $\mathbf{A}$, and hence $\mathbf{c}_{Ei}$ is the corresponding eigenvector. When G is undirected and connected, the largest eigenvector of $\mathbf{A}$ will be simple and its eigenvector will have entries that are all nonzero and share the same sign. Convention is to report the absolute values of these entries, which will automatically lie between 0 and 1 by the orthonormality of eigenvectors.

Calculation of the largest eigenvalue of a matrix and its eigenvector is a standard problem. The power method is generally used, a description of which can be found in most textbooks on computational linear algebra (e.g., Golub and van Loan [181]). This method is iterative and is guaranteed to converge under various conditions, such as when the matrix is symmetric, which $\mathbf{A}$ will be for undirected graphs. The rate of convergence to $\mathbf{c}_{Ei}$ will behave like a power, in the number of iterations, of the ratio of the second largest eigenvalue of $\mathbf{A}$ to the first. Since the power method involves only matrix-vector multiplications, it requires only $O(N_v^2)$ operations per iteration, which places it roughly on par with the other two methods described above.

Each of the three centrality measures defined so far – based on closeness, betweenness, and eigenvectors – are well motivated by particular interpretations of the term 'central'. However, as these interpretations differ from each other, it should not be surprising that the results obtained by applying them can differ accordingly.

Example 4.4 (Comparing Centrality Measures). Consider the small toy graph shown in Figure 4.4(a), consisting of just $N_v = 11$ vertices and $N_e = 12$ edges. Which vertices are more 'central' and which less? Clearly it depends to some extent upon the context in which the question is posed. Arguably the four small green vertices at the extremes of the graph (i.e., top and bottom) are not likely to be considered central under any intuitively appealing definition of the word. But it is less clear how the other vertices compare – particularly the red, yellow, and dark blue vertices.

[7] In fact, modifications of this algorithm yield similarly efficient algorithms for various other notions of betweenness centrality and for other common centrality measures, such as the closeness betweenness $c_{Cl}(v)$ defined previously. See Brandes [51, 52].

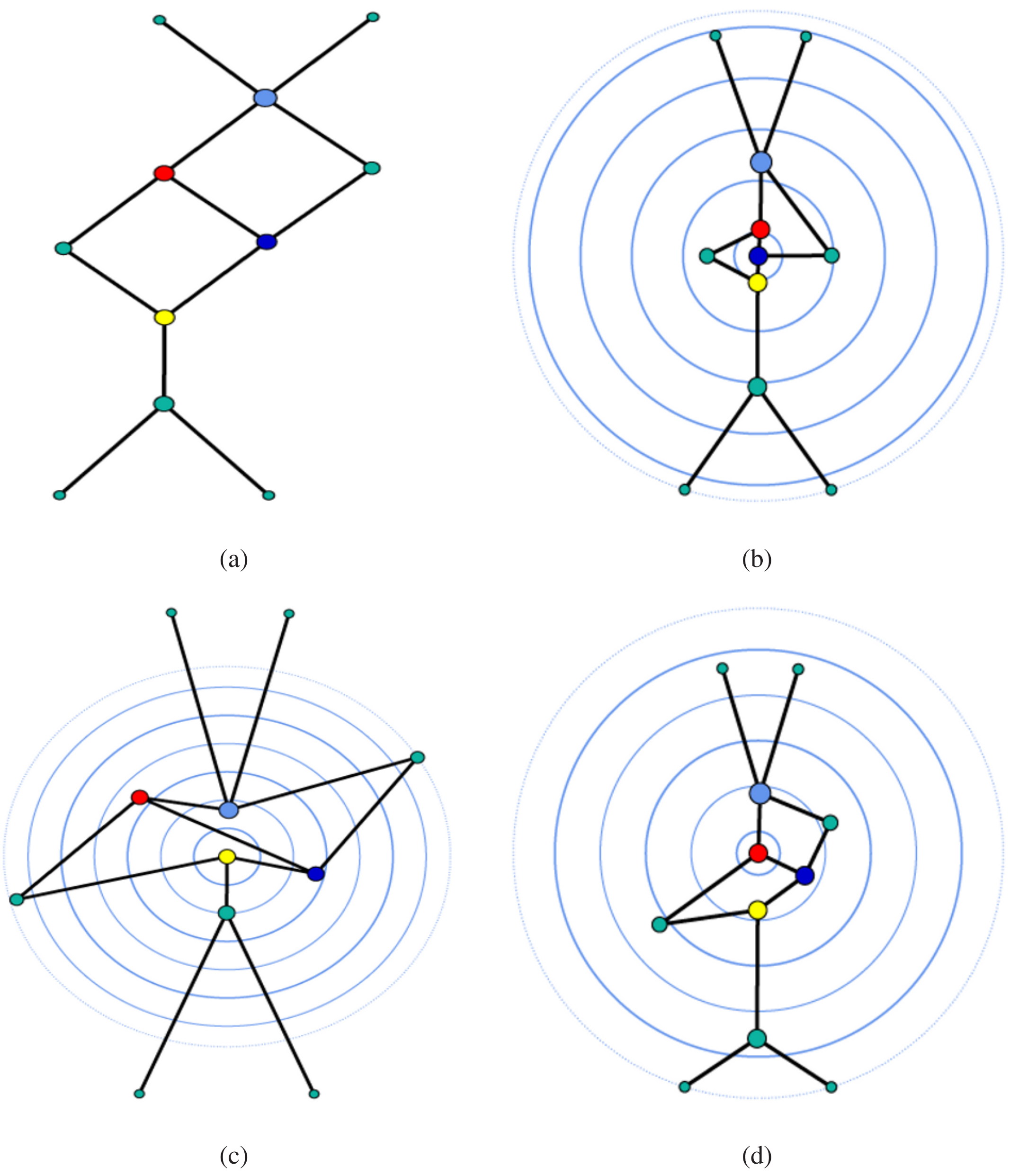

Fig. 4.4 Illustration of (b) closeness, (c) betweenness, and (d) eigenvector centrality measures on the graph in (a). Example and figures courtesy of Ulrik Brandes.

Figures 4.4(b) – (d) provide visual summaries of the closeness, betweenness, and eigenvector centralities, respectively, of the vertices in our toy graph. In each case, vertices are arranged using a radial layout, with more central vertices located closer to the center.

Under the closeness centrality, $c_{Cl}(v)$, the dark blue vertex is judged to be most central under this measure, followed closely by the red and yellow vertices. However, under the betweenness centrality, $c_B(v)$, it can be seen that the yellow vertex is now judged to be most central. In fact, note that the relative positions of all but the most extreme vertices (i.e., small green) have changed noticeably from Figure 4.4(b). Evidently, that a vertex lies close to other vertices does not necessarily mean it must also lie on many shortest paths between pairs of other vertices. The yellow and light blue vertices, as well as the large green vertex immediately beneath the yellow vertex, all essentially serve as 'gate-keepers' on paths connecting to extreme vertices, and the values $c_B(v)$ reflect this accordingly. Lastly, under the eigenvector centrality, $c_{Ei}(v)$, the relationship among vertices is again different from that which it was under the other two centrality measures, although it is arguably closer in nature to that yielded by the closeness centrality. Here the red vertex is now judged to be most central, with four of the other ten vertices sharing a similar centrality as runners up. □

4.2.2.2 Additional Measures of Centrality

Example 4.4, although a 'toy,' is far from diabolical and is intended to illustrate how even in a seemingly unremarkable graph different centrality measures can produce distinctly different orderings of the vertices. As mentioned at the start of this section, there are in fact many additional vertex centrality measures. Deciding which are most appropriate for a given application clearly requires consideration of the context.

The centrality measures defined above have all had numerous variations and extensions defined. Closeness centrality is a simple inverse function of distance on the graph. Another related idea, proposed by Hage and Harary [190], is to measure centrality as inversely proportional to the eccentricity of a vertex (i.e., the length of the longest shortest path on which it serves as an endpoint). Betweenness centrality counts all shortest paths passing through a vertex. Variations on this idea have suggested modifying the underlying path set, instead counting, for example, only shortest paths of length bounded by some k or shortest paths that are disjoint in their vertices and/or edges. See Borgatti and Everett [48] for pointers to a number of such proposals, as well as White and Smyth [402]. Finally, the notion of eigenvector centrality already had a number of very similar precursors prior to Bonacich's formulation. See Wasserman and Faust [393], for example, for background and references. Additional pointers to references and surveys may be found in Section 4.6.

Also worth special note are two fairly celebrated algorithms of a nature closely related to eigenvector centrality that have emerged from the computer science community in the context of judging the importance of web pages. The first is the PageRank algorithm (Page, Brin, Motwani, and Winograd [307]), a key component of the Google search engine (i.e., see Brin and Page [63]), which poses a model for a random 'surfing' of the Web, and produces centralities that are the solution of a rescaled and damped version of the eigensystem in earlier methods of this genre. The second is the HITS algorithm, based on the concept of 'hubs and authorities,' introduced by Kleinberg [230], which characterizes the importance of so-called hub web pages by how many authority web pages they point to, and so-called authority web pages by how many hubs point to them. Specifically, given an adjacency matrix $\mathbf{A}$ for a directed web graph, hubs are determined according to the eigenvector centrality of the matrix $\mathbf{M}_{hub} = \mathbf{A}\mathbf{A}^T$, and authorities, according to that of $\mathbf{M}_{auth} = \mathbf{A}^T\mathbf{A}$. The matrices $\mathbf{M}_{hub}$ and $\mathbf{M}_{auth}$ are analogous to quantities encountered in bibliometrics, where the former counts the number of common third works cited by a pair of works, and the latter counts the number of times two works were both cited by a third work. Both the PageRank and the 'Hubs and Authorities' algorithms have already had additional extensions and variations proposed.

There are also centrality measures deriving from the notion of flows on a network graph (e.g., Freeman, Borgatti, and White [159]), and the closely related idea of a random walk on a graph (e.g., Newman [295]). These essentially introduce another perspective on the concept of betweenness, and can be shown to have interesting connections to closeness-based centrality measures (e.g., White and Smyth [402], Brandes and Fleischer [55]).

All of the centralities discussed so far are for vertices, as it seems to be most common in practice that questions of importance are in regard to the vertices of a graph. But there are contexts where edges arguably are of greater concern than the vertices – for example, links in a physical Internet traffic network. Betweenness vertex centrality extends to edges in a straightforward manner. However, many other vertex centrality measures do not. One way around this problem is to apply vertex centrality measures to the vertices in the dual graph of a network graph G. The *dual graph* of G, say $G' = (V', E')$, is obtained essentially by changing vertices of G to edges, and edges, to vertices. More formally, the dual vertices $v' \in V'$ represent the original edges $e \in E$, and the dual edges $e' \in E'$ indicate that the two corresponding original edges in G were incident to a common vertex in G. See Chapter 3 of the edited volume of Brandes and Erlebach [54] for a brief discussion.

Lastly, we point out that for any given centrality measure $c(v)$, while it is defined on a vertex-specific (or edge-specific) basis, it may be interesting and useful in the course of an analysis to look at aggregate graph-level summaries. For instance, we can look at the distribution of the values $\{c(v)\}_{v\in V}$, in analogy to the degree distribution, and, obviously, functions of that distribution, such as moments and quantiles. In the social network literature, it has traditionally also been popular to use the *centralization index* of Freeman [157]

$$c = \frac{\sum_{v\in V}\left[c^* - c(v)\right]}{\max \sum_{v\in V}\left[c^* - c(v)\right]} \quad , \tag{4.9}$$

where c^* is the maximum of the $c(v)$ over the graph G, and the max in the denominator is over all possible graphs of order N_v. In practice, however, this quantity is not easily calculated for most $c(v)$, outside of certain special cases.

As an intermediate choice, between single-vertex and whole-graph centrality measures, Everett and Borgatti [139] offer extensions of vertex degree, closeness, and betweenness centralities to groups of arbitrary size. See also Kolaczyk, Chua, and Barthélemy [233], for additional work on group-level extensions of vertex betweenness centrality, and Puzi, Elovici, and Dolev [316], who provide a fast algorithm for successive computation of group betweenness.

4.3 Characterizing Network Cohesion

A great many questions in network analysis boil down to questions involving *network cohesion*, the extent to which subsets of vertices are cohesive – or 'stuck together' – with respect to the relation defining edges in the network graph. Do friends of a given actor in a social network tend to be friends of one another as well? What collections of proteins in a cell appear to work closely together? Does the structure of the pages in the World Wide Web tend to separate with respect to distinct types of content? What portion of a measured Internet topology would seem to constitute the 'backbone'?

There are many ways that we can define network cohesion, depending on the context of the question being asked. Definitions differ, for example, in scale, ranging from local (e.g., triads) to global (e.g., giant components), and also in the extent to which they are specified explicitly (e.g., cliques) versus implicitly (e.g., 'clusters'). In this section we discuss a number of common ways to define and summarize 'cohesion' in a network graph.[8]

4.3.1 Local Density

Many notions of coherent network structure are based on the principle that a coherent subset of nodes should be locally 'dense' in the graph. From a graph-theoretic point of view, one immediately obvious concept to employ in this regard is that of a *clique* – a complete subgraph H of G. Cliques are subsets of vertices that are fully cohesive, in the sense that all vertices within the subset are connected by edges. A case of common practical interest, particularly in social network analysis, is that of 3-cliques (i.e., triangles). In practice, large cliques are relatively rare, as they necessarily require that G itself be fairly dense. For example, Turán [384] shows that a sufficient condition for a clique of size n to exist in G is that $N_e > (N_v^2/2)[(n-2)/(n-1)]$. This condition may be contrasted with the fact that real-world networks generally have been found to have N_e closer to the same order as N_v.

There are many questions that we might ask involving cliques in a graph, but it should be noted that the corresponding algorithms for calculating answers can range in computational complexity from linear to NP-complete. For instance, both the determination of whether a specific subset of nodes $U \subseteq V$ is a clique and whether it is maximal can be done in $O(N_v + N_e)$ time. On the other hand, the problem of determining whether a graph G has a maximal clique of at least size n is NP-complete. Certain other problems lie between these two extremes. For example,

[8] One could also arguably use the term 'group', in its colloquial sense, in reference to the various types of cohesive network sub-structures we seek to characterize with the tools in this section. However, in the field of social network analysis, from whence many of the tools in this chapter come, the notion of a social group has certain more precise connotations. In order to avoid confusion in this regard, we have chosen to use the somewhat more formal sounding term 'cohesion.'

detecting all triangles (i.e., all cliques of size $n = 3$) in G can be done exhaustively in $O(N_v^3)$ time, which can be improved to $O(N_v^{2.376})$ time in general and $O(N_e^{1.41})$ time in sparse graphs through use of the adjacency matrix $\mathbf{A}$ of G and fast algorithms for multiplication of square matrices. See Chapter 6 of the edited volume of Brandes and Erlebach [54] for additional details on these and similar results.

Ultimately, cliques tend to be an overly restrictive definition of network cohesion, and therefore are not used extensively in practice in and of themselves. Instead, weakened versions of this idea tend to be more practical. One such notion is that of *plexes*. A subgraph H consisting of m vertices is called an n-plex, for $m > n$, if no vertex has degree less than $m - n$. In other words, we specify that no vertex is missing more than n of its possible $m - 1$ edges with the other vertices in the subgraph. Unfortunately, as intuitively appealing as this variation may be, from a computational perspective problems involving n-plexes tend to scale like those involving cliques.

Much more computationally tractable is a dual notion of cohesiveness, that of *cores*, the utility of which we have already seen demonstrated in Chapter 3 in the context of the router-level Internet. Cores also have found successful usage in problems involving protein interaction networks, including the search for molecular complexes (Bader and Hogue [15]), the prediction of protein function (Altaf-Ul-Amin et al. [7]), and the study of evolutionary conservation (Wuchty and Almaas [408]). Recall from Section 3.5.2.3 that a k-core of a graph G is a subgraph H for which all vertices have degree at least k. A maximal k-core subgraph may be computed in as little as $O(N_v + N_e)$ time (Batagelj, Mrvar, and Zaversnik [25]). In fact, the algorithm computes the shell indices for every vertex in the graph, where the shell index of $v \in V$ is the largest value, say c, such that v belongs to the c-core of G but not its $(c+1)$-core. The efficiency of the algorithm is due to iterative use of (i) the observation that, for a given vertex, those neighbors with lesser degree lead to a decrease in the potential shell index of that vertex, and (ii) an efficient technique for sorting vertices by degree.

All of the approaches to network cohesion above proceed by first stating a prespecified notion of locally dense structure and then looking to see whether it is anywhere satisfied in a graph G and, if so, where. Alternatively, we can instead define a measure of local density and then characterize the extent to which subsets of vertices are dense, according to this measure. Such measures are commonly based on ratios of the number of edges among a subset of vertices to the total number of possible edges. For example, in a graph G with no self-loops and no multiple edges, the *density* of a subgraph $H = (V_H, E_H)$ is

$$\mathsf{den}(H) = \frac{|E_H|}{|V_H|(|V_H| - 1)/2} \quad . \tag{4.10}$$

The value of $\mathsf{den}(H)$ will lie between zero and one and provides a measure of how close H is to being a clique. Note that $\mathsf{den}(H)$ is just a rescaling of the average degree $\bar{d}(H)$ of H, since $\mathsf{den}(H) = (|V_H| - 1)\bar{d}(H)$.

There is freedom in the choice of subgraph H defining (4.10). Taking $H = G$ yields the density of the overall graph G. Conversely, taking $H = H_v$ to be the set of neighbors of a vertex $v \in V$, and the edges between them, yields a measure of density in the immediate neighborhood of v. Watts and Strogatz [396] propose $\mathsf{den}(H_v)$ as a summary of the extent to which there is 'clustering' of edges local to v and propose that the average of $\mathsf{den}(H_v)$ over all vertices be used as a *clustering coefficient* for the overall graph.

These measures of clustering can be expressed alternatively in terms of the density of triangles among connected triples. A *triangle* is a complete subgraph of order three. A *connected triple* is a subgraph of three vertices connected by two edges (also sometimes called a 2-star). Intuitively, a measure of the frequency with which connected triples 'close' to form triangles will provide some indication of the extent to which edges are 'clustered' in the graph. Let $\tau_\triangle(v)$ denote the number of triangles in G into which $v \in V$ falls, and $\tau_3(v)$, the number of connected triples in G for which the two edges are both incident to v. Note that $\tau_3(v) = \binom{d_v}{2}$. The Watts-Strogatz local clustering coefficient $\mathsf{den}(H_v)$ can be re-expressed as $\mathsf{cl}(v) = \tau_\triangle(v)/\tau_3(v)$, for those vertices v with $\tau_3(v) > 0$. The corresponding clustering coefficient for G takes the form[9]

$$\mathsf{cl}(G) = \frac{1}{V'} \sum_{v \in V'} \mathsf{cl}(v) \quad , \tag{4.11}$$

where $V' \subseteq V$ is the set of vertices v with $d_v \geq 2$.

The value $\mathsf{cl}(G)$ is a popular choice of clustering coefficient in the literature. However, as Bollobás and Riordan [45] have pointed out, being an average of averages, this quantity may be less informative than the appropriate weighted average, which takes the form

$$\frac{\sum_{v \in V'} \tau_3(v)\mathsf{cl}(v)}{\sum_{v \in V'} \tau_3(v)} \quad . \tag{4.12}$$

The quantity (4.12) in turn is equal to

$$\mathsf{cl}_T(G) = \frac{3\tau_\triangle(G)}{\tau_3(G)} \quad , \tag{4.13}$$

where $\tau_\triangle(G) = (1/3)\sum_{v \in V} \tau_\triangle(v)$ is the number of triangles in the graph, and $\tau_3(G) = \sum_{v \in V} \tau_3(v)$, the number of connected triples. The value $\mathsf{cl}_T(G)$, called the *transitivity* of the graph, is a standard quantity in the social network literature, where it is also referred to as the 'fraction of transitive triples.' Transitivity in this context refers to, for example, the case where the friend of your friend is also a friend of yours.

Note that, while the two clustering coefficients cl and cl_T often yield similar values for a graph G, they are not equivalent, and in fact can differ noticeably in some cases. For example, in the karate club network of Chapter 1, where it is easy to see many triads and triangles, the clustering coefficient values are $\mathsf{cl}(G) = 0.5879$

[9] There is some variation in the literature in defining this quantity, depending on the treatment of vertices v with degree zero or one. See also Barrat and Weigt [19].

and $\mathsf{cl}_T(G) = 0.2557$. It is even possible to define highly imbalanced graphs so that $\mathsf{cl}(G) \to 1$ while $\mathsf{cl}_T(G) \to 0$, as N_v grows. See Bollobás and Riordan [45, pg. 18], or the volume edited by Brandes and Erlebach [54, Ch. 11]. In practice, therefore, it is important to clearly state which clustering coefficient is being reported.

The key requirement for calculating either of these clustering coefficients is an ability to identify connected triples $\{v, u_1, u_2\}$ at each vertex $v \in V$ and to check whether the pairs $\{u_1, u_2\}$ form an edge in E. These steps are inherent in the algorithms for enumerating the triangles in a graph, already described above, and hence the complexity for computing these clustering coefficients scales accordingly, using these algorithms.

Clustering coefficients have become a standard quantity used in the analysis of network structure. Interestingly, their values have typically been found to be quite large in real-world networks, in comparison to what otherwise might be expected based on classical random graph models.[10] In large-scale networks with broad degree distributions, it has frequently been found that the local clustering coefficient $\mathsf{cl}(v)$ varies inversely with vertex degree. See Newman [296, Sec. III.B], for a number of references.

Higher-order clustering coefficients have also been proposed, involving cycles of length greater than three (i.e., recall that the triangles, upon which cl and cl_T are based, are 3-cycles). For example, Caldarelli, Pastor-Satorras, and Vespignani [72] and Newman [290] consider the use of four-cycles, and Fronczak, Hołyst, Jedynak, and Sienkiewicz [163] consider arbitrary k-cycles.

4.3.2 Connectivity

In contrast to a local perspective, and the search for small-scale subsets of cohesive vertices, we may also find it useful in some contexts to take a larger, global perspective. A basic question of interest is whether a given graph separates into distinct subgraphs. If it does not, we might seek to quantify how close to being able to do so it is. Intimately related to such issues are questions associated with the flow of 'information' in the graph.

4.3.2.1 Connected Components and 'Small Worlds'

Recall that a graph G is said to be connected if every vertex is reachable from every other (i.e., if for any two vertices, there exists a walk between the two). A *connected component* of a graph is a maximally connected subgraph.

In practice, the process of constructing a network graph, as described in Chapter 3, may not necessarily result in a single connected graph. Rather, the graph

[10] That is, for example, as compared to a graph where edges are assigned to pairs of vertices randomly through independent coin flips with common probability $p \in (0,1)$. See Section 6.2.1.

may consist of a number of separate connected components. The task of verifying whether a graph is connected and, if not, identifying its connected components, can be done in $O(N_v + N_e)$ time, using either depth-first search (DFS) or breadth-first search (BFS) algorithms. See the text by Cormen, Leiserson, Rivest, and Stein [100], for example.

Often it is the case that one of the connected components in a graph G dominates the others in magnitude, in that it contains the vast majority of the vertices in G. Such a component is called, borrowing terminology from random graph theory, the *giant component*. In the case of an unconnected graph with a giant component, depending on the task at hand, it may be sensible to restrict attention to that component alone in carrying out further analysis and modeling. For example, the network of protein interactions described in Section 4.2.1.1 has a giant component with 5,128 vertices, and 11 smaller components, one with three vertices, and 10 with two vertices. An algorithm that seeks to predict the biological function of a given protein, based on knowledge of the functions of its neighbors in this network, such as those described in Chapter 8, is most sensibly applied to the giant component.

A celebrated characteristic observed in the giant component of many real-world networks is the so-called *small world* property. The initial concept and terminology for this term is usually traced back to the now-famous experiment of Stanley Milgram [277], in the late 1960's, and his assertion that people are only separated by roughly six acquaintances (i.e., by 'six degrees of separation,' as in the play of Guare [188] of the same name). That is, if we were to represent the population of the world and acquaintances among its members as a social network graph, despite its enormous size, the typical number of 'hops' along shortest paths between its vertices would be quite small.

Formally, in analyzing a network graph, the term 'small' here is usually meant to indicate the average distance between distinct vertices, say

$$\bar{l} = \frac{1}{N_v(N_v+1)/2} \sum_{u \neq v \in V} \mathsf{dist}(u,v) \quad , \tag{4.14}$$

scales as $O(\log N_v)$ or less. For example, the protein interaction network would be considered to have the small-world property in this sense, as there is an average distance of only 3.68 among the 5,128 vertices in its giant component; in fact, the maximum distance is only 9. The discovery that such a property in fact holds for many types of networks is attributed to Watts and Strogatz [396]. These authors also found that often such small average distance $\bar{l}$ is accompanied by a high clustering coefficient cl or cl_T, and they proposed a random graph model for such networks – see Section 6.3. It is now formally in the sense of these two properties holding jointly that the term 'small world' is used, although informally it is still often used to refer to small average distance alone.

It should not be altogether surprising – at least in hindsight – that networks might have such a property, as the following informal argument shows. If each vertex in the network graph has roughly d immediate neighbors, then we expect to find roughly d^2 neighbors within $h = 2$ hops of any vertex, d^3 within $h = 3$ hops, and so on. With this exponentially increasing size of successive neighborhoods, we soon expect to reach a value h^* for which the expression $d^{h^*} \approx N_v$ holds. From this it can be seen that $h^* \approx \log N_v / \log d$ should scale logarithmically in N_v. Of course, this argument is simplistic in assuming an essentially homogeneous distribution of vertex degrees and ignoring any effects due to variations in local density. Nevertheless, the end result remains valid in more complicated models, as we will discuss in Section 6.3.

On the other hand, surprising or not, it is clear that the ‘small world’ property has potentially important implications on a host of fundamental problems relating to networks and ‘communication’ upon them, whether referring to the exchange of gossip in a social network, the search for content in a World Wide Web network, or the exchange of an infectious disease in an epidemiological network. We will return to this topic in our discussion of dynamic processes on networks in Chapter 8.

4.3.2.2 Vertex/Edge-Connectivity, Cuts, and Flows

A somewhat more refined notion of connectivity derives from asking questions like, “If an arbitrary subset of k vertices (edges) is removed from a graph, is the remaining subgraph connected?” Picture, for example, the extreme case where G consists of two cliques joined by a single edge between a vertex in each. A useful image here is that of a ‘barbell’ – two masses joined by a single, straight rod. Intuitively, we would be inclined to think of G as consisting of ‘nearly two components.’ The concepts of vertex- and edge-connectivity, and the related concepts of vertex- and edge-cuts, help to make these notions precise.

A graph G is called *k-vertex-connected* if (i) $N_v > k$, and (ii) the removal of any subset of vertices $X \subset V$ of cardinality $|X| < k$ leaves a subgraph $G - X$ that is connected. Note that, for graphs G with at least two vertices, G is 1-vertex-connected if and only if it is connected. Similarly, G is called *k-edge-connected* if (i) $N_v \geq 2$, and (ii) the removal of any subset of edges $Y \subseteq E$ of cardinality $|Y| < k$ leaves a subgraph $G - Y$ that is connected. The *vertex (edge) connectivity* of G is the largest integer such that G is k-vertex- (k-edge-) connected. It can be shown that the vertex connectivity is bounded above by the edge connectivity, which in turn is bounded above by the minimum degree d_{min} among vertices in G.

Standard questions of interest involving these quantities include: “Is G k-vertex- (k-edge-) connected?” “What is the vertex(edge)-connectivity of G?” and “What are the maximal k-vertex- (k-edge-) connected components of G?” Algorithms exist for answering these questions that run in polynomial time. For example, the vertex-connectivity (edge-connectivity) of a graph can be calculated in time that runs like the connectivity k times N_v^2 (i.e., somewhere between quadratic and cubic time). See the volume edited by Brandes and Erlebach [54, Ch. 7], for example.

A fundamental result in graph theory, known as Menger’s theorem, essentially states that a nontrivial graph G is k-vertex (k-edge) connected if and only if all pairs of distinct vertices $u, v \in V$ can be connected by k vertex-disjoint (edge-disjoint) paths. This result relates the robustness of a graph in the face of removal of its vertices (edges) to the richness of distinct paths running throughout it. A graph with low vertex (edge) connectivity therefore can have the paths, and hence any 'information' flowing over those paths, disrupted by removing an appropriate choice of a correspondingly small number of vertices (edges).

If the removal of a particular set of vertices (edges) in G actually disconnects the graph, that set is called a *vertex-cut (edge-cut)*. Sometimes it is of interest to find cut sets that are in some sense 'minimal.' For a given pair of vertices $u, v \in V$, a *u-v-cut* is a partition of V into two disjoint, non-empty subsets $S, \bar{S} \subset V$, where $u \in S$ and $v \in \bar{S}$. For G equipped with edge weights w_e, a u-v-cut is said to be a *minimum u-v-cut* if the sum of the weights on edges connecting vertices in S to vertices in $\bar{S}$ is a minimum. If the edge weights are all of unit value (i.e., $w_e \equiv 1$), then finding a minimum u-v-cut is equivalent to finding an edge-cut of minimal cardinality, with one component containing u and the other containing v. Note that if the smallest cardinality of such minimum edge-cut sets, over all distinct $u, v \in V$, is say k, then clearly the edge-connectivity of G is $k-1$.

To find a minimum u-v-cut, the problem typically is translated to an equivalent problem of maximizing a measure of flow on the edges of a derived directed graph, subject to certain constraints, by invoking the Max-Flow Min-Cut theorem of Ford and Fulkerson [146]. There are many algorithms that have been developed for solving the max-flow problem, all of which run in polynomial time. The algorithm of Goldberg and Tarjan [177], for example, runs in $O(N_v N_e \log(N_v^2/N_e))$ time. To find a minimum cut over all pairs of vertices it is sufficient to make only $N_v - 1$ calls to a max-flow algorithm (Gomory and Hu [182]). Alternatively, more efficient would be to use the non-flow-based algorithm of Stoer and Wagner [369], which runs in $O(N_e N_v + N_v^2 \log N_v)$.

4.3.2.3 Connectivity in Directed Graphs

Most of the concepts introduced above, and their corresponding algorithmic implementations, extend to the case of directed graphs in a straightforward manner. Recall from Chapter 2, for example, that a digraph G is *weakly connected* if its underlying graph (i.e., the result of stripping away the labels 'tail' and 'head' from G) is connected, and that it is called *strongly connected* if every vertex v is reachable from every other vertex u by a directed walk. The concept of vertex-connectivity (edge-connectivity) extends analogously by requiring that the graphs remaining after removal of vertices (edges) be strongly connected. Similarly, the notion of cuts and flows remain essentially unchanged, except that now there is a directionality designated. For example, in a cut $(S, \bar{S})$, we would refer to one of the sets, say S, as the *source*, and other, $\bar{S}$, as the *sink*, suggestive of the movement of flows from S to $\bar{S}$.

An often-useful characterization of directed network graphs is that of a 'bowtie,' shown schematically in Figure 4.5. The bowtie structure was first described by Broder and colleagues [67], in the context of the World Wide Web graph. This structure consists of five parts. The first part is a strongly connected component (SCC). The second is an *in-component*, which consists of vertices from which the

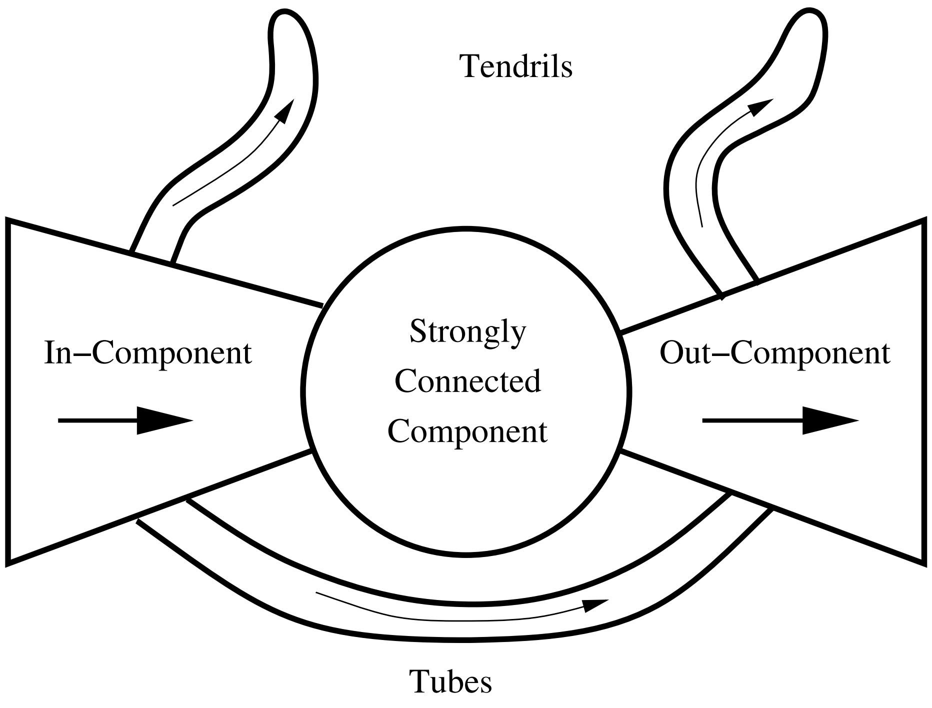

Fig. 4.5 'Bowtie' structure of a directed network graph. Adapted from Broder et al. [67].

SCC can be reached, but which themselves cannot be reached from the SCC. Third is an analogous *out-component*, whose vertices can be reached from the SCC but from which the SCC cannot be reached. The in- and out-components are in some sense 'upstream' and 'downstream' from the SCC, respectively, and are thus typically drawn to the left and right of the SCC, as in Figure 4.5. The fourth and fifth parts of the bowtie structure are the *tubes* and *tendrils*. The former is composed of vertices between the in- and out-components that are not part of the SCC, while the latter consists of vertices that can neither reach nor be reached from the SCC.

Example 4.5 (Bowtie Structure of the AIDS Blog Network). Figure 4.6 shows a bowtie decomposition of the AIDS blog network from Chapter 1. It can be seen that the SCC is quite small (four vertices), as is the in-component (two vertices). The network is dominated by the out-component (112 vertices), and a few tendrils (28 vertices). Note that in large-scale networks, the digraph G may be disconnected. So the SCC may in fact be a giant strongly connected component (GSCC), with corresponding giant in- and out-components. In the context of measurements on the World Wide Web graph – vastly larger than the tiny blog network shown here – Broder and colleagues found that the relative size of these four parts of the bowtie in their giant component were in fact roughly equal in size. □

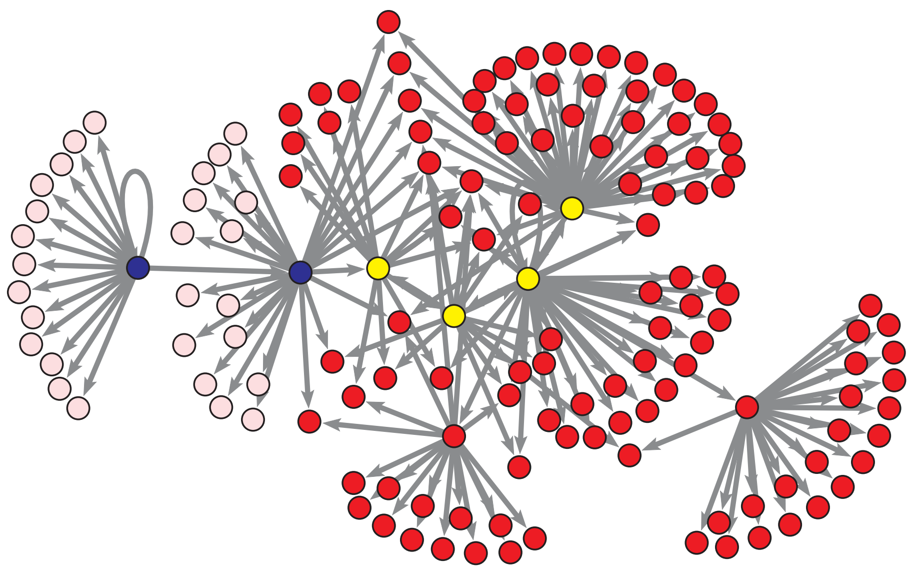

Fig. 4.6 AIDS blog network, from Chapter 1, with nodes colored according to their membership in the 'bowtie' decomposition: strongly connected component (yellow), in-component (blue), out-component (red), and tendrils (pink).

4.3.3 Graph Partitioning

Partitioning – broadly speaking – refers to the segmentation of a set of elements into 'natural' subsets. More formally, a *partition* $\mathscr{C} = \{C_1, \ldots, C_K\}$ of a finite set S is a decomposition of S into K disjoint, nonempty subsets C_k such that $\cup_{k=1}^{K} C_k = S$. The need for partitioning arises in various scientific disciplines, in the context of problems ranging from image segmentation to pattern recognition to information retrieval. In the analysis of network graphs, partitioning is a useful tool for finding, in an unsupervised fashion, subsets of vertices that demonstrate a 'cohesiveness' with respect to the underlying relational patterns. It has been used, for example, in the detection of community structure in social networks (e.g., Girvan and Newman [174]) and in the identification of possible protein complexes from protein interaction networks (e.g., Sen, Kloczkowski, and Jernigan [346], and the many references therein).

A 'cohesive' subset of vertices generally is taken to refer to a subset of vertices that (i) are well connected among themselves, and at the same time (ii) are relatively well separated from the remaining vertices. More formally, graph partitioning algorithms typically seek a partition $\mathscr{C} = \{C_1, \ldots, C_K\}$ of the vertex set V of a graph $G = (V, E)$ in such a manner that the sets $E(C_k, C_{k'})$ of edges connecting vertices in C_k to vertices in $C_{k'}$ are relatively small in size compared to the sets $E(C_k) = E(C_k, C_k)$ of edges connecting vertices within the C_k.

While there are a number of standard methods of graph partitioning in use, the development of new methods is nevertheless also a very active area of research, a fact that is at least in part suggestive of the inherent difficulty of the problem. We will describe two well-established classes of methods – those based on adaptations of hierarchical clustering and those based on principles of spectral partitioning – in some detail, and then briefly discuss a handful of other methods, including a number of those proposed more recently.

4.3.3.1 Hierarchical Clustering

A great many methods for graph partitioning are essentially variations on the more general concept of *hierarchical clustering* used in data analysis.[11] There are numerous techniques that have been proposed for the general clustering problem, differing primarily in (i) how they evaluate the quality of proposed clusterings and (ii) the algorithms by which they seek to optimize that quality. See Jain, Murty, and Flynn [213], for example. These methods take a greedy approach to searching the space of all possible partitions $\mathscr{C}$, by iteratively modifying successive candidate partitions. Hierarchical methods are classified as either *agglomerative*, being based on the successive coarsening of partitions through the process of merging, or *divisive*, being based on the successive refinement of partitions through the process of splitting. At each stage, the current candidate partition is modified in a way that minimizes a specified measure of cost. In agglomerative methods, the least costly merge of two previously existing partition elements is executed, whereas in divisive methods, it is the least costly split of a single existing partition element into two that is executed.

Whether agglomerative or divisive, when used for network graph partitioning, hierarchical clustering methods actually produce, as the name indicates, an entire hierarchy of nested partitions of the graph, not just a single partition. These partitions can range fully between the two trivial partitions $\{\{v_1,\},\ldots,\{v_{N_v}\}\}$ and V. Agglomerative methods begin with the former of these two, while divisive methods begin with the latter. The resulting hierarchy typically is represented in the form of a tree, called a *dendrogram*, such as that shown in Figure 4.7.

The measure of cost incorporated into a hierarchical clustering method used in graph partitioning should reflect our sense of what defines a 'cohesive' subset of vertices. It also can have important implications on the computational complexity of the overall algorithm implementing the method. At the core of most of these cost functions is a measure of (dis)similarity, say x_{ij}, between pairs of vertices $v_i, v_j \in V$. Methods differ not only in how the (dis)similarity between vertex pairs is defined,

[11] Note that 'clustering' as used here, which is standard terminology in the broader data analysis community, differs from 'clustering' as used in Section 4.3.1, which arose in the social network community, in reference to the coefficients cl and cl_T used to summarize the relative density of triangles among connected triples.

but also in how (dis)similarity between sets of vertices is defined. For example, in agglomerative algorithms, given two sets of vertices C_1 and C_2, two standard approaches to assigning a similarity value to this pair of sets is to use the maximum (called *single-linkage*) or the minimum (called *complete linkage*) of the x_{ij}, over all pairs $v_i \in C_1$ and $v_j \in C_2$. Other approaches can be more involved. For example, the default method in the software package `Pajek` is based on a method similar to that of Ward [391]. Ward's method utilizes group sums of squares, but is intended for Euclidean data. The `Pajek` implementation involves an extension of this method to non-Euclidean data, as described in Batagelj [22].

Example 4.6 (Hierarchical Clustering of the Karate Club Network). The dendrogram in Figure 4.7 shows the results of an agglomerative analysis of the karate club network shown in Figure 1.2, and described in Section 1.2.2. These results are based on a dissimilarity x_{ij} for vertices v_i and v_j defined as

$$x_{ij} = \frac{\left|\mathscr{N}_{v_i} \triangle \mathscr{N}_{v_j}\right|}{d_{(N_v)} + d_{(N_v - 1)}} \quad , \tag{4.15}$$

where $\mathscr{N}_v$ is the set of neighbors of a vertex v, '$\triangle$' indicates symmetric difference of two sets,[12] and $d_{(i)}$ is the i-th smallest element in the degree sequence. In other words, x_{ij} is just the number of neighbors of v_i and v_j that are not shared, normalized to lie in the interval $[0,1]$, so that 0 and 1 indicate perfect similarity and dissimilarity, respectively. Dissimilarities for subsets of vertices were calculated from the x_{ij} using the extension of Ward's method mentioned above, and the lengths of the branches in the dendrogram are in relative proportion to the changes in dissimilarity with successive splits. Generally, interest is in partitions immediately following a comparatively long set of branches. With this particular network, we know that the karate club eventually split into two separate clubs. The first partition, into just two subsets of vertices, separates the network precisely according to the eventual breakup. □

There are various other common choices of (dis)similarity measures, such as the so-called 'Euclidean distance' dissimilarity

$$x_{ij} = \sqrt{\sum_{k \neq i,j} \left(A_{ik} - A_{jk}\right)^2} \quad . \tag{4.16}$$

Here $\mathbf{A}$ is the adjacency matrix, and x_{ij} measures the standard Euclidean distance between rows i and j. Hierarchical clustering algorithms based on (dis)similarities of this sort are reasonably efficient, running in $O(N_v^2 \log N_v)$ time. Presuming the N_v^2 (dis)similarity values can be computed in $O(N_v^2)$ time at most, as is the case on

[12] The *symmetric difference* of two sets S_1 and S_2 is the set of elements that are in one or the other but not both, $S_1 \triangle S_2 = (S_1 - S_2) \cup (S_2 - S_1)$.

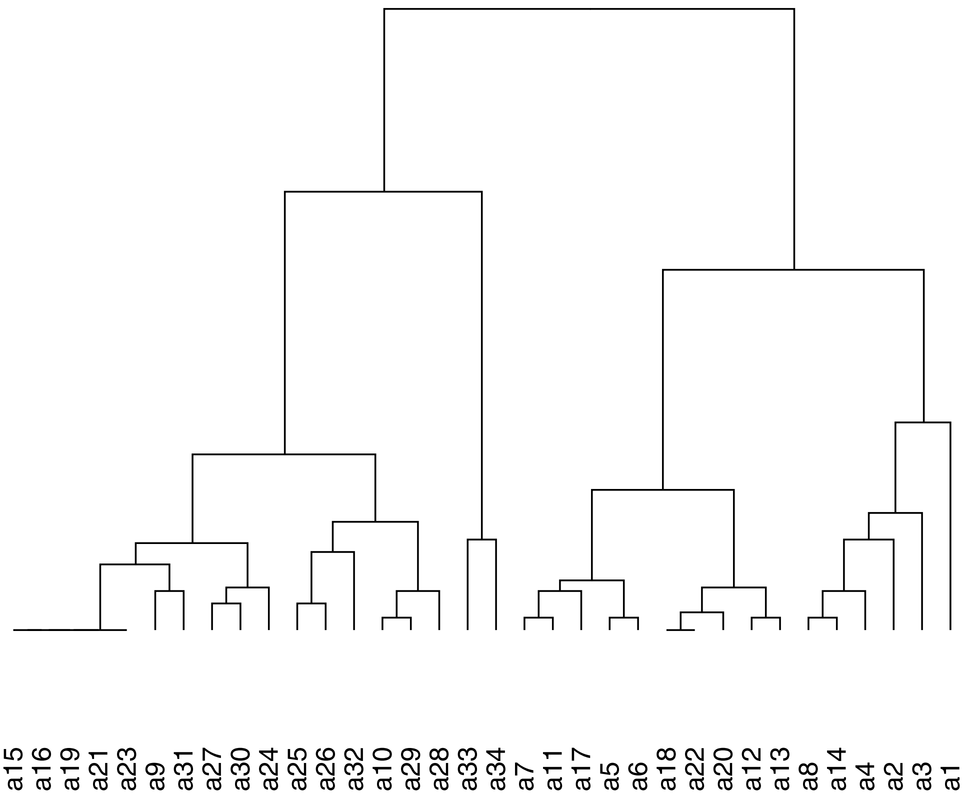

Fig. 4.7 Hierarchical clustering of the karate club network.

sparse graphs for (4.15) and (4.16), the cost is dominated by the operation of sorting these values.[13]

We note that not all hierarchical clustering methods employ vertex-by-vertex (dis)similarity measures. For example, an agglomerative approach proposed by Newman [294] seeks to greedily optimize the so-called *modularity* of a partition. Let $\mathscr{C} = \{C_1, \ldots, C_K\}$ be a given candidate partition and define $f_{ij} = f_{ij}(\mathscr{C})$ to be the fraction of edges in the original network that connect vertices in C_i with vertices in C_j. The modularity of $\mathscr{C}$, as defined by Newman and Girvan [297], is the value

$$\mathsf{mod}(\mathscr{C}) = \sum_{k=1}^{K} \left[f_{kk}(\mathscr{C}) - f_{kk}^* \right]^2 \ , \tag{4.17}$$

where f_{kk}^* is the expected value of f_{kk} under some model of random edge assignment. Most commonly, f_{kk}^* is defined to be $f_{k+}f_{+k}$, where f_{k+} and f_{+k} are the k-th row and column sums of $\mathbf{f}$, the $K \times K$ matrix[14] formed by the entries f_{ij}. This choice corresponds to a model in which a graph is constructed to have the same degree distribution as G, but with edges otherwise placed at random, without respect to

[13] For sorting n objects, good algorithms typically run in $O(n \log n)$ time.

[14] Note that for undirected graphs this matrix will be symmetric, and hence $f_{k+} = f_{+k}$; for directed graphs, however, $\mathbf{f}$ can be asymmetric.

the underlying partition elements dictated by $\mathscr{C}$. Large values of the modularity are therefore taken to suggest that $\mathscr{C}$ captures nontrivial 'group' structure, beyond that expected to occur under the random assignment of edges.

Note that in this approach, because a single common quality measure $\mathsf{mod}(\mathscr{C})$ is applied to all candidate partitions $\mathscr{C}$, as opposed to using an *ad hoc* propagation of (dis)similarities, we have a natural mechanism not only for generating the hierarchy but also for choosing an optimal partition within that hierarchy. Optimization of the modularity function (4.17) can be shown to be an *NP*-complete problem (Brandes et al. [61]). A greedy agglomerative algorithm is therefore commonly used for heuristic optimization (Newman [294], Clauset, Newman, and Moore [92]). It runs in $O(N_v^2 \log N_v)$ time generally, using sophisticated data structures, but nearly linear time on sparse networks with hierarchical structure, making it appealing for use in the context of certain large networks. See Clauset, Newman, and Moore [92] and Brandes et al. [61] for details. In practice, this algorithm has been found to frequently yield useful results. Other methods of a similar nature are described in the volume edited by Brandes and Erlebach [54, Ch. 8].

4.3.3.2 Spectral Partitioning

Another common approach to graph partitioning is to exploit results in spectral graph theory that associate the connectivity of a graph G with the eigen-analysis of certain matrices, such as the adjacency matrix or the Laplacian. Here we review two popular approaches to spectral partitioning. For general background on spectral graph theory, see the monograph by Chung [90].

The first approach is to conduct a spectral analysis of the adjacency matrix $\mathbf{A}$ of a graph. That is, we begin by calculating the collection of N_v pairs $(\lambda_i, \mathbf{x}_i) \in \mathbb{R} \times \mathbb{R}^{N_v}$ for which $\mathbf{A}\mathbf{x}_i = \lambda_i \mathbf{x}_i$, where $\lambda_1 \leq \ldots \leq \lambda_{N_v}$ denote the N_v (not necessarily distinct) eigenvalues of $\mathbf{A}$, ordered from smallest to largest, and $\mathbf{x}_1, \ldots, \mathbf{x}_{N_v}$, their corresponding eigenvectors. Then, starting with the largest (absolute) eigenvalues, the entries of each eigenvector $\mathbf{x}_i$ are sorted, and the vertices corresponding to particularly large positive or negative entries, in conjunction with their immediate neighbors, are declared to be a cluster. Note that in practice attention is usually restricted to just the K-th largest eigenvalue/vector pairs, for $K \sim \log N_v$.

The motivation behind this approach is the fact that, in the case of a graph G consisting of two d-regular graphs joined to each other by just a handful of vertices, the two largest eigenvalues will be roughly equal to d, and the remaining eigenvalues will be of only $O(d^{1/2})$ in magnitude. That is, there will be a gap in the spectrum of eigenvalues – a 'spectral gap.' In addition, the two eigenvectors corresponding to the largest two eigenvalues can be expected to have large positive entries on vertices of one sub-network and large negative entries on vertices of the other sub-network, with the two eigenvectors emphasizing the vertices in each sub-network in a complementary fashion. See Gkantsidis, Mihail, and Zegura [175], for example, for a summary of these and related results, in the context of a spectral analysis of Internet topology data.

Example 4.7 (Spectral Analysis of the Karate Club Network). Figure 4.8 shows the eigenvalues, and the first two eigenvectors, from a spectral analysis of the karate club network graph. We see that the first two eigenvalues are fairly distinct from the rest, indicating the possible presence of two sub-graphs. The first eigenvector appears to suggest a separation of the 1st and 34th actors, and some of their nearest neighbors, from the rest of the actors. The second eigenvector in turn provides evidence that these two actors, and certain of their neighbors, should themselves be separated. Actors 1 and 34 represent the director and administrator, respectively, of the original club. Together, these two eigenvectors go some ways towards separating the two clubs into which the actors eventually separated. □

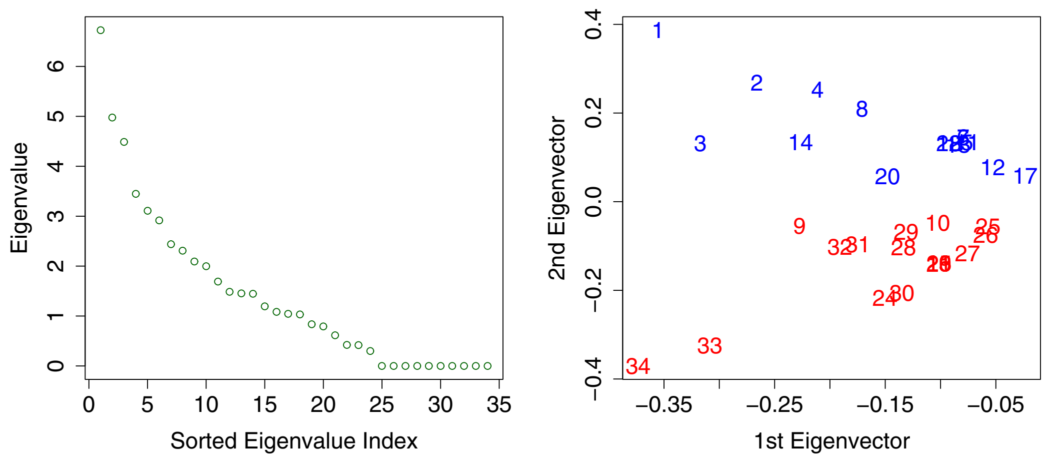

Fig. 4.8 Spectral analysis of the karate club network. Left: the (absolute) eigenvalues. Right: the first two eigenvectors, where points are labeled according to the number of their corresponding actors, and colored according to the sub-group of each actor, as displayed in Figure 1.2.

In reality, the method is known not to work as well when the sub-networks are far from regular and instead the network has a broad degree distribution, as is more common in practice. In addition, it should be noted that the partitions found through spectral analysis will tend to be ordered and separated by vertex degree, since the eigenvalues will mirror quite closely the underlying degree distribution. This has been observed empirically – it can be seen in the analysis of the karate club network above, for example – and has been proved rigorously for various network models like those we will see in Chapter 6. See Gkantsidis, Mihail, and Zegura [175] for some discussion, as well as references to a handful of formal results. Of course, this close relationship between partitioning and degree may not necessarily be seen as a problem. Sen, Kloczkowski, and Jernigan [346], for example, argue that such results are sensible in the context of protein complexes, based on biological evidence. Where it is suspected to be problematic, in yielding partitions that are unrepresentative of the true underlying 'group' structure, normalizing the adjacency matrix to have unit row sums is a commonly proposed solution (e.g., Gkantsidis, Mihail, and Zegura [175]).

The other popular approach to spectral partitioning involves the iterative bisection of a network graph according to spectral properties of the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D} = \mathrm{diag}[(d_v)]$. A formal result in spectral graph theory states that a graph G will consist of K connected components if and only if $\lambda_1(\mathbf{L}) = \cdots = \lambda_K(\mathbf{L}) = 0$ and $\lambda_{K+1}(\mathbf{L}) > 0$. See Godsil and Royle [176, Lemma 13.1.1]. The smallest eigenvalue of $\mathbf{L}$ can be shown to be identically equal to zero, with corresponding eigenvector $\mathbf{x}_1 = (1,\ldots,1)^T$. Therefore, if we suspect a graph G to consist of 'nearly' $K = 2$ components, for example, and hence be amenable to bisection, we should expect $\lambda_2(\mathbf{L})$ to be close to zero.

Interestingly, this value λ_2 can be closely associated with a number of measures of graph connectivity and structure. See the survey by Mohar [281] or, more recently, the volume edited by Brandes and Erlebach [54, Ch. 14]. Among these, of particular relevance to the clustering problem is the *isoperimetric number* of a graph,

$$\phi(G) = \min_{S \subset V : |S| \le N_v/2} \phi(S,\bar{S}) \ , \tag{4.18}$$

where $\phi(S,\bar{S}) = |E(S,\bar{S})|/|S|$ is called the *ratio* of the cut defined by $(S,\bar{S})$. Recalling our discussion of vertex- and edge-cuts in Section 4.3.2, it is clear that the ratio is a natural quantity to minimize in seeking a good bisection of G. Unfortunately, this minimization problem is *NP*-hard. But $\phi(G)$ can be bounded as

$$\frac{\lambda_2}{2} \le \phi(G) \le \sqrt{\lambda_2(2d_{max} - \lambda_2)} \ , \tag{4.19}$$

where d_{max} is the maximum degree of G. Thus $\phi(G)$ will be small whenever λ_2 is small and vice versa. See Brandes and Erlebach [54, Ch. 14].

Fiedler [144], the first to associate λ_2 with the connectivity of a graph, suggested partitioning vertices by separating them according to the sign of their entries in the corresponding eigenvector $\mathbf{x}_2$, producing a cut of the form

$$S = \{v \in V : \mathbf{x}_2(v) \ge 0\} \quad \text{and} \quad \bar{S} = \{v \in V : \mathbf{x}_2(v) < 0\} \ . \tag{4.20}$$

The vector $\mathbf{x}_2$ is hence often called the *Fiedler vector*, and the corresponding eigenvalue λ_2, the *Fiedler value*. The latter is also known as the *algebraic connectivity* of the graph. Since it can be shown (see Mihail [276]) that

$$\phi(G) \le \phi(S,\bar{S}) \le \frac{\phi^2(G)}{d_{max}} \le \lambda_2 \ , \tag{4.21}$$

where $\phi(S,\bar{S})$ is the ratio of the cut in (4.20), spectral bisection can be viewed as a computationally efficient approximation to finding a best cut achieving $\phi(G)$.

Example 4.8 (Spectral Bisection of the Karate Club Network). Figure 4.9 shows a plot of $\mathbf{x}_2$ for the karate club network, in which it can be seen that spectral bisection

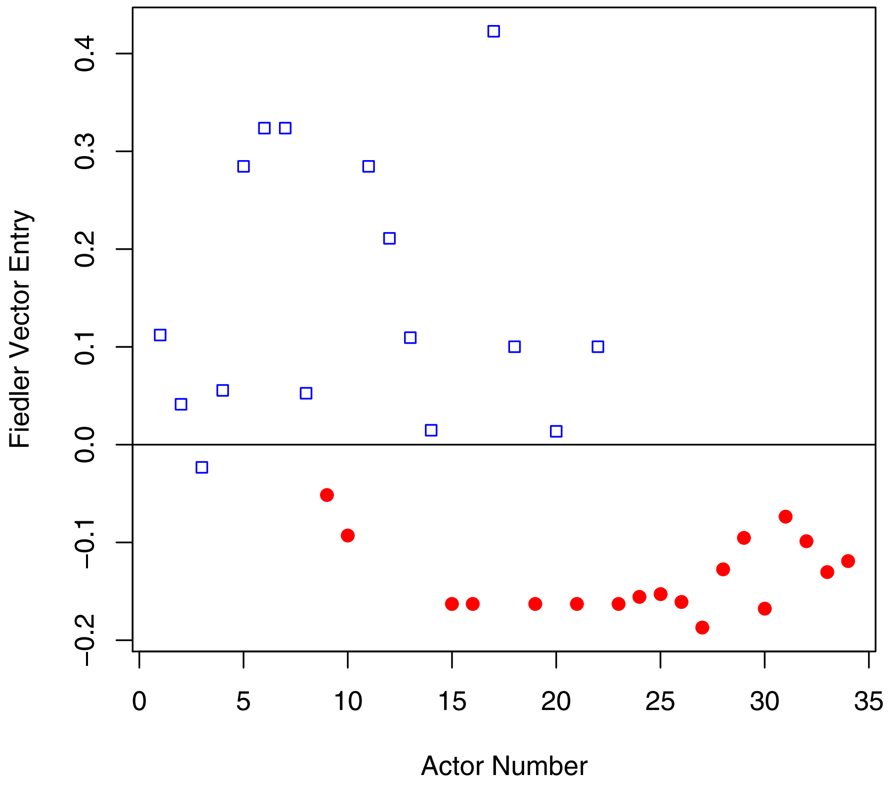

Fig. 4.9 Plot of the Fiedler vector, $\mathbf{x}_2$, for the karate club network graph. The horizontal line at $y = 0$ indicates the clustering suggested by spectral bisection. Color and shape of the symbols for individual actors correspond to those of Figure 1.2.

produces a nearly perfect partition of the graph, classifying all but the third actor correctly. In practice, the bisection operation is often applied recursively, first to the graph G as a whole, and then within the subgraphs corresponding to each of the two partition elements identified, and so on. It can be shown rigorously that such an approach works well on bounded-degree planar graphs and certain mesh graphs (Spielman and Teng [366]), although in other contexts it cannot always be expected to give satisfactory results. Here, obviously, continuing to bisect the first two partition elements indicated by $\mathbf{x}_2$ will lead us farther away from the two-group structure known to exist in this network. $\square$

For both of the spectral partitioning methods described here, the computational overhead is in principle determined by the cost of computing the eigen-decomposition of an $N_v \times N_v$ matrix, which takes $O(N_v^3)$ time. However, realistically, only a small subset of extreme eigenvalues, and their corresponding eigenvectors, are needed in each case. Furthermore, the matrices $\mathbf{A}$ and $\mathbf{L}$ will typically be quite sparse in practice. Therefore, spectral analyses of this type, for sparse graphs as large as tens of thousands of vertices and hundreds of thousands of edges, can be implemented efficiently using variants of the iterative Lanczos algorithm, which is specifically designed for such settings. In the case of spectral bisection, for example, the $O(N_v^3)$ time required by the nominal algorithm is reduced to $O(N_e/(\lambda_3 - \lambda_2))$, which will behave almost linearly in the number of edges N_e if the gap between the second and third smallest eigenvalues is sufficiently large (i.e., when the graph G really does nearly separate into two components). See Golub and van Loan [181] for additional details on algorithms of this type.

4.3.3.3 Additional Methods of Graph Partitioning

There are many other methods for partitioning graphs that have been proposed in various literatures – far too many to survey here. Instead, we restrict ourselves to describing just a handful of additional methods, most of which at least partly involve variations or extensions of principles underlying the two basic methods already described above.

For example, a small body of literature has emerged around a proposal of Girvan and Newman [174], involving the iterative removal of edges in a graph. The premise underlying the proposed method is that certain edges in a network graph may carry more of the ‘flow’ in that graph than others, and that it is these edges that likely serve as ‘bridges’ separating ‘cohesive’ subsets of vertices. Using the edge-betweenness centrality mentioned at the end of Section 4.2.2 to quantify the level of flow over an edge, the authors propose a divisive hierarchical algorithm that removes the edge of largest betweenness, recomputes the betweenness of all remaining edges, and continues iteratively until no more edges remain.

This method seems to work quite well in practice, but can be somewhat costly from a computational perspective, requiring $O(N_v^3)$ time. There have been several suggestions for speeding up the basic algorithm (e.g., see Newman [289] for discussion and references), including a parallel implementation proposed by Yang and Lonardi [409] that has yielded nearly linear empirical run times. Of course, another approach to the cost problem – and of interest in its own right – is the substitution of other measures than edge-betweenness to prioritize the removal of edges. Measures based on concepts of resistance and random walks in networks are examined in Newman and Girvan [297], for example, although both of these choices are in fact more computationally costly than edge-betweenness.

Newman [294] has observed that the hierarchical clustering algorithm described earlier in this section, based on the modularity in (4.17), seems to yield results similar to those produced by such edge-removal algorithms. The use of modularity as a general measure of quality for proposed clusterings (i.e., distinct from its use with the agglomerative algorithm proposed in Newman [294]) has itself received some attention in the graph partitioning community, particularly among those interested in the detection of communities in social networks. With respect to the computation of modularity, there have been suggestions for approximate optimization of modularity using spectral methods. For example, Newman [291] proposes a method whose technical development parallels that of the spectral bisection method quite closely, but with a matrix related to modularity playing the role of the Laplacian L. Similarly, White and Smyth [402] have proposed a method of optimization based on spectral relaxation.

Another notable use of concepts from spectral analysis may be found in the method of Gibson, Kleinberg, and Raghavan [172], who characterize community structure among pages in the World Wide Web. There the eigenvectors of the pair of matrices $\mathbf{M}_{hub} = \mathbf{A}\mathbf{A}^T$ and $\mathbf{M}_{auth} = \mathbf{A}^T\mathbf{A}$ are used to assign vertices membership in clusters, where A is the adjacency matrix of a directed graph G. The matrices $\mathbf{M}_{auth}$ and $\mathbf{M}_{hub}$ were seen to underlie the hubs-and-authorities notion of centrality mentioned in Section 4.2.2.2.

Also no doubt deserving mention here is the general body of techniques referred to as *block modeling*. First proposed by White, Boorman, and Breiger [401], techniques of this sort seek to model the relationships among 'positions' in a network, and often bear a strong algorithmic resemblance to clustering methods. Because the motivation for and details of work in this area has tended to be strongly intertwined with various notions of social structure, to do them justice is beyond the scope of this book. We instead refer the reader to the book by Doreian, Batagelj, and Ferligoj [122]. See also Scott [344], Wasserman and Faust [393], and the volume edited by Brandes and Erlebach [54, Chs. 9 and 10], who in addition provide information on the closely related topics of positional and role analysis.

Finally, as a side note, it is perhaps worth pointing out that it is not uncommon to see methods designed for cluster analysis in general datasets directly applied to the context of graphs. Typically the graph is first embedded into a Euclidean space, using a tool like multi-dimensional scaling (as underlies the Kamada-Kawai layout algorithm discussed in Section 3.4.2), and then a standard clustering algorithm, like k-means clustering, is applied to the 'data' that are the embedded vertices, using the corresponding Euclidean distances.

4.3.4 Assortativity and Mixing

As has been mentioned a number of times so far in this section on the characterization of network cohesion, it is generally expected, where cohesive subsets of vertices are present in a network graph, that underlying these subsets there is some commonality in certain relevant characteristics (or attributes) of the vertices. For example, proteins found to cluster in a graph of protein interactions often participate in similar biological processes; similarly, actors found to cluster in a social network may share similar relevant interests. Graph partitioning methods are useful precisely because these characteristics will often be unobserved (i.e., they will be latent). However, conversely, given certain measured vertex characteristics, it can be useful to quantify the extent to which vertices may be partitioned according to them.

Selective linking among vertices, according to a certain characteristic(s), is termed *assortative mixing* in the social network literature, and measures that quantify the extent of assortative mixing in a given network have been referred to as *assortativity coefficients*. The assortativity coefficients we describe here are attributed to Newman [293] and are essentially variations on the concept of correlation coefficients.[15]

To begin, we note that the vertex characteristics involved could be categorical, ordinal, or continuous. Consider the categorical case, and suppose that each vertex in a graph G can be labeled according to one of M categories. Let f_{ij} be the fraction of edges in G that join a vertex in the i-th category with a vertex in the j-th category; denote the i-th marginal row and column sums of the resulting matrix $\mathbf{f}$ by f_{i+} and f_{+i}, respectively. Note that these quantities are defined in complete analogy to the same quantities that underlie our definition of modularity (4.17). We then define the assortativity coefficient r_a to be

$$r_a = \frac{\sum_i f_{ii} - \sum_i f_{i+} f_{+i}}{1 - \sum_i f_{i+} f_{+i}} \quad . \tag{4.22}$$

The value r_a is equal to zero when the mixing in the graph is no different from that obtained through a random assignment of edges that preserves the marginal degree distribution (see the related discussion surrounding (4.17) above). Similarly, it is equal to one when there is perfect assortative mixing (i.e., when edges only connect vertices of the same category). When the mixing is perfectly disassortative, in the sense that every edge in the graph connects vertices of two different categories, then the coefficient in (4.22) takes a minimum value of

$$r_a^{min} = -\frac{\sum_i f_{i+} f_{+i}}{1 - \sum_i f_{i+} f_{+i}} \quad . \tag{4.23}$$

Hence, the value r_a lies between -1 and 1, but does not achieve -1 in the case of perfect disassortative mixing. Newman [293] argues that this behavior may be interpreted as reflecting that disassortative networks can range less 'far' from randomness than assortative networks.

Example 4.9 (Assortativity in the Abilene Network). Consider the Abilene network in Figure 1.1, described in Section 1.2.1. A quick glance at the network graph in this figure is sufficient to strongly suggest the presence of disassortative mixing. The Abilene network, like the Internet as a whole, is dominated by a clear hierarchical structure. There are 11 'core nodes' in the network. The core nodes are linked both to each other and to additional nodes of certain types. However, most of these latter nodes serve only to join Abilene with other Internet networks. And neither these nodes nor the other networks tend to connect with each other directly. Table 4.1

[15] Newman's ideas in turn extend a popular assortativity coefficient originally proposed by Gupta, Anderson, and May [189]. However, the latter is somewhat problematic in that it can allow arbitrarily small subsets of vertices to have a disproportionately large effect. See Newman [293, App. A] for discussion.

shows the actual numbers of category-to-category edges in this network, for the six categories of nodes defined in the figure. The f_{ij} are obtained by scaling these values by the total 675. The assortativity coefficient is found to be $r_a = -0.3162$, which is not far from the minimum value $r_a^{min} = -0.3461$, thus strongly supporting our suspicions of disassortative mixing. □

	Core	Exchange	Peer	Connector	Participant	Conn./Part.
Core	14	6	5	17	0	16
Exchange	6	1	46	2	0	0
Peer	5	46	0	0	0	1
Connector	17	2	0	0	203	0
Participant	0	0	0	203	0	34
Conn./Part.	16	0	1	34	34	0

Table 4.1 Counts of edges between vertices of six different types in the Abilene network depicted in Figure 1.1.

The extension of these ideas to ordinal or continuous characteristics is straightforward. An example, in the context of social networks, of a vertex characteristic of this type for which there might be assortative mixing, is the age of the actor corresponding to each vertex. See Newman [293, Fig 1], for example. Note that age can be treated as either continuous or, if rounded, ordinal. For a given characteristic of interest, denote by (x_e, y_e) the values of that characteristic for the vertices joined by an edge $e \in E$. Then a natural candidate for quantifying the assortativity in this characteristic is just the Pearson correlation coefficient of the pairs (x_e, y_e),

$$r = \frac{\sum_{x,y} xy\,(f_{xy} - f_{x+}f_{+y})}{\sigma_x \sigma_y} \quad . \tag{4.24}$$

Here the summation is over all unique observed pairs (x,y), with f_{xy}, f_{x+}, and f_{+y} defined in analogy to the categorical case, and σ_x and σ_y are the standard deviations corresponding to the distribution of frequencies $\{f_{x+}\}$ and $\{f_{+y}\}$, respectively. In the case where the characteristic of interest is vertex degree, note that we simply recover the degree correlation described in Section 4.2.1.

On a related note, we mention that the similar structure underlying the definition of the modularity $\mathrm{mod}(\mathscr{C})$ of a partitioning $\mathscr{C}$ in (4.17), on the one hand, and the assortativity coefficients r_a and r in (4.22) and (4.24), on the other, allow for interesting connections to be made between modularity and assortativity, as discussed in Newman [291, Sec. VIII.A].

4.4 Case Study: Analysis of an Epileptic Seizure

Epilepsy is a chronic neurological condition and the world's most common serious brain disorder. It is characterized by seizures that are associated with abnormal activity among neurons in the brain. Epilepsy is commonly controlled through medication, but in those cases where medication no longer prevents seizures, surgical removal of the affected zone of the brain may be necessary. Perhaps somewhat surprisingly, this surgical procedure has not changed substantially since its inception in the late 19th century, although improved imaging and analysis techniques – including network analysis techniques – are helping to refine it. In order to illustrate the integrated usage of some of the concepts and tools introduced in this chapter, we present a network-oriented analysis of epileptic seizure data in humans, following Kramer, Kolaczyk, and Kirsch [239].

Figure 4.10 shows the nodes in an 8×8 electrode grid that was implanted in the cortical surface of the brain of a patient with epilepsy. Accompanying this grid were two strips (not shown) of six electrodes each, implanted deeper into the brain. Together, these implants allow for the collection of electrocorticogram (ECoG) data, which consist of time series of voltage levels in the vicinity of each of the 76 electrodes. The voltage levels, in turn, are indicative of the levels of local brain activity. An example of one such time series is shown in the figure. Also indicated are two separate portions of this series, corresponding to 10-second intervals identified as being prior to and immediately after the start of a seizure in the patient, referred to as the 'preictal' and 'ictal' periods, respectively. Our interest will be in comparing the activity of the brain during these two periods.

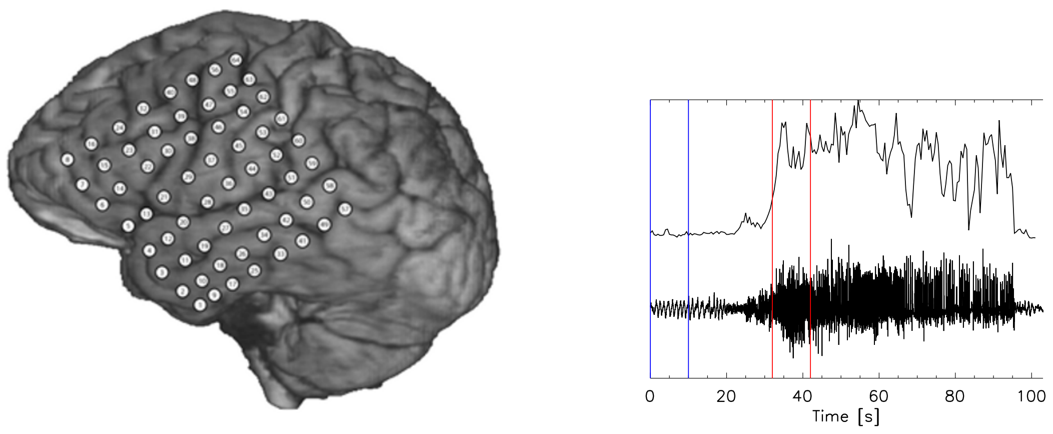

Fig. 4.10 Left: Three-dimensional reconstruction of the brain of an epilepsy patient, with ECoG implant grid super-imposed. Right: Example of an ECoG time series at one electrode, both for one seizure (bottom) and after smoothing (i.e., bandpass filtering) and averaging over eight such seizures (top). Preictal (blue) and ictal (red) periods are indicated between pairs of parallel lines.

Network graphs are a natural way to represent the interactive activity – or coupling – among different regions of the brain. The review of Boccaletti et al. [41]

includes a short survey of various examples of such usage of network graphs, at different scales and based on different types of measurements. Here, network graphs were generated to summarize the cortical-level coupling among the different measured regions of the brain, for both preictal and ictal periods. Specifically, within each period, for each pair among the 76 electrodes, a measure of correlation was computed between the two corresponding 10-second time series, a threshold was applied, and an edge was assigned to the vertices representing these electrodes if their correlation exceeded that threshold. This process was carried out for data on eight different seizures in this patient, measured over five consecutive days, essentially following Kramer, Kolaczyk, and Kirsch [239].

An example of the resulting network graphs, corresponding to one seizure, is shown in Figure 4.11. A mere glance at the two graphs in the figure is enough to see that they strongly suggest that the brain is in two very different states prior to and during the seizure. In particular, there appears to be a thinning of edges throughout the network, indicating a reduced level of coupling between different cortical regions in the brain. This behavior is especially apparent in the bottom corner of the grid, closest to the two strips of electrodes, which in turn were located close to where the seizure was suspected to emanate.

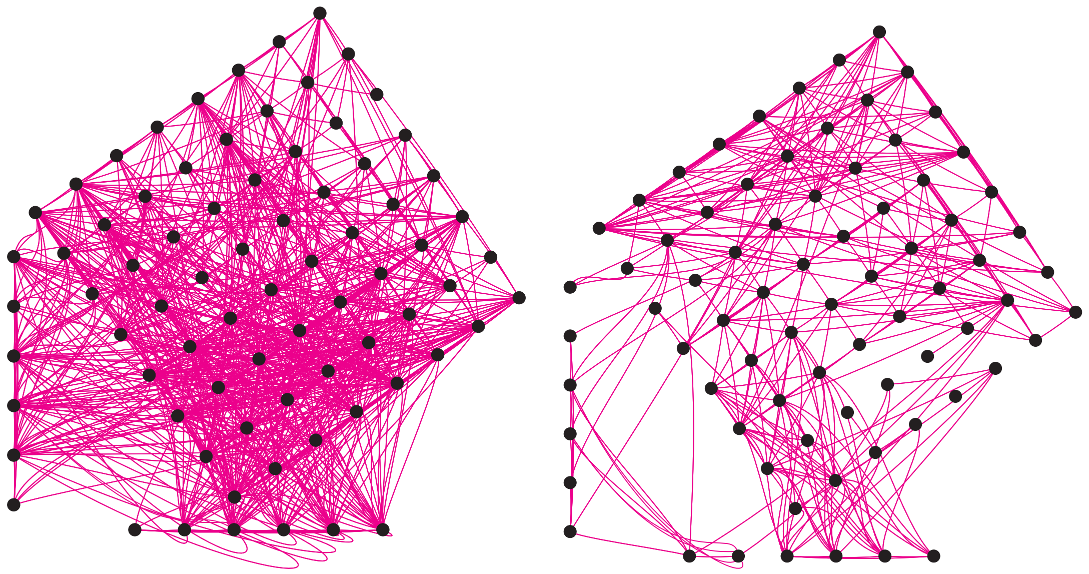
Fig. 4.11 Network representations of cortical-level coupling between brain regions about each electrode, during preictal (left) and ictal (right) periods.

A number of the summaries of network characteristics described in this chapter are useful for quantifying the apparent preictal-to-ictal changes. For example, Figure 4.12 shows visual representations of degree, closeness and betweenness centrality, and clustering coefficient values for each of the 76 vertices in our network. Preictal and ictal periods, as well as their difference, are shown separately, averaged over the eight seizures at each vertex. We can see, looking at the plots of vertex degree, that our initial impression of a global decrease in coupling, with a particular emphasis in the bottom corner, is confirmed. The plots of closeness centrality show an accompanying global increase in distance (i.e., decreased closeness) between vertices, again with rather greater changes in the bottom corner of the grid. Betweenness centrality is seen to have changed as well, but in a much more irregular manner. There are both increases and decreases in vertex betweenness, with some vertices having experienced particularly large increases. Note, however, that most of these vertices with large increases in betweenness experienced relatively little change in degree. Finally, both positive and negative changes are also observed in the vertex clustering coefficients, though the relative magnitudes are a bit smaller than for the other summary measures. The more pronounced changes are decreases in local clustering, and many of these seem to occur close to regions where betweenness changed the most.

Together these results suggest that for areas where local density decreases upon seizure onset, and for which a region nearby remains coupled at roughly pre-seizure levels, there is an increased burden in the form of more paths 'squeezing' through. Note that since there are eight seizures from the same patient at our disposal, treating these seizures as replicates allows for formal assessment of statistical significance of the changes noted above.[16] For example, changes in global quantities like average degree, average path length, closeness and betweenness centralization, and average clustering coefficient all are found statistically significant at the 0.05 level using paired t-tests on seven degrees of freedom. Similarly, the use of such tests on a vertex-by-vertex basis, with control for multiple testing, allows us to pinpoint the locations of the most significant contributions to these changes. For instance, at a false discovery rate (FDR) of $q = 0.05$, the 10 vertices marked with stars in Figure 4.12 are identified as locations of significant change in betweenness. Kramer, Kolaczyk, and Kirsch [239] suggest that such locations may potentially serve to more precisely guide surgical intervention.

4.5 Characterizing Dynamic Network Graphs

We address briefly here the problem of characterizing dynamic network graphs, that is, of characterizing a collection $\{G^{(t)}\}$ of network graphs indexed over times t in some set $\mathbb{T}$, as introduced in Section 3.6. The development of a comprehensive body of tools for such characterization, and of a corresponding understanding of their properties and usage, currently lags far behind that for static graphs G that has been the focus of most of this chapter. This lacking is probably due in part to insufficiently pressing need, in that traditionally it has been notably more difficult in most network studies to obtain time-indexed network data that are rich in

[16] Up until now, we have mentioned little in the way of how to assess statistical significance in the summary measures introduced in this chapter, because when given only a single network graph (or pair of graphs, in the context of our epilepsy case study), such assessments necessarily must rely on network graph models, such as we will see in Chapter 6. Here, however, the availability of repeated observations allows us to assess significance based on sample variability.

both quality and quantity. The situation, however, is beginning to change, particularly where Internet-based measurement and archival resources can be leveraged, as in Leskovec, Kleinberg, and Faloutsos [255] or Kossinets and Watts [238]. Also contributing to the gap between static and dynamic methods of characterization is possibly the absence of a mature graph-theoretic infrastructure. For example, an extension to dynamic networks of Menger's fundamental theorem (relating paths and cut-sets) was only proposed relatively recently, by Kempe, Kleinberg, and Kumar [224]. Finally, another important factor is the sheer combinatorial explosion in problem variants, once the factor of 'time' is to be incorporated. See Moody, McFarland, and Bender-deMoll [285], for example, for a detailed discussion of how and why such variants arise, in the context of social networks.

There are various representations of a dynamic network with which we might work, roughly paralleling those found in traditional statistical time series analysis. For example, we might study cumulative versions of the network, in the sense that a vertex or edge present in $G^{(t)}$ is present in all $G^{(t')}$, for $t' > t$ (e.g., Leskovec, Kleinberg, and Faloutsos [255]). Or rather we might study discrete 'snapshots,' where each $G^{(t)}$ features only those vertices and edges present in a certain interval of time around t. Depending on the temporal resolution of the measurements in such cases, some authors have found it useful to work with a smoothed sequence of graphs across time, based on, for example, moving windows (e.g., Kossinets and Watts [238]) and moving averages (e.g., Hill, Agarwal, Bell, and Volinksy [198]). In some cases, it may be possible to obtain network information continuous in time, which can allow for analyses not possible with measurements at other temporal resolutions (e.g., Kumar, Raghavan, Novak, and Tomkins [242], Priebe, Conroy, Marchette, and Park [315]).

The potential effects of different representations of a dynamic network on the analysis of its structural characteristics is presumably an important issue, but one that has received little formal attention to date. In studies analyzing dynamic network graphs, two issues that have received fairly consistent attention are (i) the evolution over time of standard static global graph properties, such as degree characteristics, diameter, and clustering behavior, and (ii) the dynamics of the underlying formation and dissolution of edges (or arcs, in the directed case) among vertices.

With respect to the evolution of standard global properties, an important question is whether various properties are stable in time. In Leskovec, Kleinberg, and Faloutsos [255], for example, it is demonstrated that – contrary to current wisdom – the network graphs for a number of large-scale systems (e.g., citations and coauthorship in various literatures, and autonomous systems in the Internet) experience both an increase in density and a shrinking in diameter over time. On the other hand, Kossinets and Watts [238] found that many characteristics of a large email-based social network of university students and faculty remained fairly stable over time, outside of expected periods of fluctuation, like summer and winter recess.

Regarding the dynamics of the network edges themselves, this is a topic that a subset of researchers in social network analysis have concerned themselves with for some time, particularly with respect to the role of potential social mechanisms. See the edited volume by Doreian and Stokman [121], for example, and the references therein. However, most studies in that area have revolved around datasets for rather small social groups, which makes rigorous empirical assessment of proposed mechanisms and models difficult. Datasets across a range of scales – both in terms of network size and in terms of longitudinal time points – are needed to help gain additional insight into potentially non-trivial dynamic process structures. Furthermore, information beyond just vertex and edge lists can be expected to be necessary, such as that provided by longitudinal measurements on appropriate vertex characteristics (e.g., group affiliations of social actors). The email-based social network in Kossinets and Watts [238], for example, involved roughly 14.5 million exchanges within a university population of about $43,500$ people, over a period of 355 days, as well as various personal attributes on each individual. Such data allow for use of, for instance, formal methods of statistical survival analysis in assessing the effect of vertex characteristics on the rates at which triadic closure (i.e., triads turning into triangles) occurs.

When continuous-time data are available, it is possible to ask questions about changes in graphs on very short time scales. For example, Kumar, Raghavan, Novak, and Tomkins [242] present an analysis of large-scale blogging activity in the World Wide Web in which such continuous-time data allowed for the discovery of short, dense 'bursty' periods of the creation of edges within communities. Similarly, Priebe, Conroy, Marchette, and Park [315] introduce methods for the detection of change-point anomalies in graph structure, illustrated in the context of the network of email correspondence surrounding the Enron scandal.

4.6 Additional Related Topics and Reading

The description and summary of structural characteristics of an appropriate network graph is frequently an important and necessary first step in the analysis of network data, and often it serves as the primary part of such analyses. The array of tools and techniques proposed for this purpose is quite large and still growing. Here in this chapter we have contented ourselves simply with presenting material on topics arguably at the core of this area.

From this vantage, the interested reader has numerous additional avenues that might be explored. The volume edited by Brandes and Erlebach [54], cited frequently throughout this chapter, provides a much more detailed – yet highly readable – presentation of most of the topics in this chapter, as well a number of others not covered here, from the perspective of computer science and applied graph theory. In the statistical physics literature, there have been a number of comprehensive review articles written from the perspective of that discipline, including that of Newman [296] and, more recently, that of Boccaletti et al. [41]. Both of these articles cover materials related both to this chapter and, particularly in the case of Boccaletti et al. [41], to later chapters on network modeling (Chapter 6) and network processes (Chapter 8). In the literature on social networks, two canonical references are the small volume by Scott [344] and the much larger volume by Wasserman and

Faust [393]. Lastly, we point out that there are nontrivial points of tangency between work on methods of structural analysis in network graphs and work on graph-based approaches to high-dimensional data analysis (e.g., manifold learning). As both areas of research are quite active currently, this is likely to be a fertile interaction to watch.[17]

Exercises

4.1. Consider the problem of estimating the exponent α for a distribution with density proportional to $x^{-\alpha}$.

a. A random variable X is said to follow a Pareto distribution if it has a density of the form

$$f(x) = \frac{\alpha x_0^{\alpha}}{x^{\alpha+1}} \ , \tag{4.25}$$

for $x > x_0$, where $x_0, \alpha > 0$. Suppose that we observe an independent and identically distributed sample $x_1, \ldots, x_n$ according to (4.25), and let x_0 be known. Show that under the Pareto model the log-likelihood

$$\ell(\alpha) = \sum_{i=1}^{n} \log\alpha + \alpha \log x_0 - (\alpha+1)\log x_i$$

is maximized by the estimator in (4.4). That is, show that the Hill estimator is the maximum likelihood estimator for α under this model.

b. Use the computer to generate independent and identically distributed samples $x_1, \ldots, x_n$ from a Pareto distribution,[18] for various choices of α and n. For each choice of these values, estimate α using the Hill estimator, and compare the performance of this estimator to that of the naive regression estimators motivated by the relation in (4.2) and the logarithm of the relation in (4.3).

4.2. Let G be an undirected, connected graph, with weights $\{w_e\}_{e\in E}$. Recall the definition of the betweenness centrality $c_B(v)$ in (4.7). The key result underlying the algorithm of Brandes [51] for computing betweenness centrality is a recursive expression for the partial sums

$$\delta_{s+}(v) = \sum_{t\in V} \delta_{st}(v) \ ,$$

where $\delta_{st}(v) = \sigma(s,t|v)/\sigma(s,t)$ is the *pair-dependency* of $s,t \in V$.

---

[17] See, for example, *http://www.ipam.ucla.edu/programs/gss2005/* .

[18] Samples may be drawn from a Pareto distribution, with parameters $x_0, \alpha > 0$ by using the inverse transform method (e.g., see Liu [264, Ch. 2.1]). Specifically, having generated (pseudo)random numbers $u_1, \ldots, u_n$ uniformly on $(0,1)$, the values $x_i = x_0/u_i^{1/\alpha}$ will follow the desired Pareto distribution.

Define the set of *predecessors* of v on shortest paths from s as

$$P_s(v) = \{u \in V : \{u,v\} \in V, \mathsf{dist}(s,v) = \mathsf{dist}(s,u) + w(u,v)\} \ ,$$

where dist is the geodesic distance and $w(u,v)$ is the weight on the edge incident to vertices $u, v \in V$.

a. Suppose that there is exactly one shortest path from $s \in V$ to each $t \in V$. Show that

$$\delta_{s+}(v) = \sum_{z: v \in P_s(z)} (1 + \delta_{s+}(z)) \ .$$

b. Show that in the general case,

$$\delta_{s+}(v) = \sum_{z: v \in P_s(z)} \frac{\sigma(s,v)}{\sigma(s,z)} \cdot (1 + \delta_{s+}(z)) \ .$$

4.3. Recall the two definitions of clustering coefficient cl and cl_T in equations (4.11) and (4.13), respectively.

a. Verify the relation between cl and cl_T in equation (4.12).

b. Construct an example of a class of graphs for which cl can be made arbitrarily close to one while cl_T becomes arbitrarily close to zero. (Hint: Consider graphs composed of a set of 'nested' triangles.)

4.4. Recall the use of the graph Laplacian $\mathbf{L}$ for spectral graph clustering, described in Section 4.3.3.

a. Suppose that G is composed of $K = 2$ separate connected components, say G_1 and G_2. Then $\mathbf{L}$ will be a block-diagonal matrix with two blocks. Recall that the smallest eigenvalue λ_1 is equal to zero, and that it corresponds to the eigenvector $\mathbf{x}_1 = (1, \ldots, 1)^T$. Show that in addition the next two smallest eigenvalues will also be zero, with degenerate eigenvectors $\mathbf{x}_k$ defined so that $x_{k,i} = 1$ if vertex i is in G_k, and zero otherwise.

b. Generate graphs G that deviate from the scenario of strictly separated components G_1 and G_2, underlying part (a), to increasingly greater extents. For example, you might let G_1 and G_2 be two identical cliques of size n, and add an increasingly larger number of edges between randomly selected pairs of vertices, one in G_1 and the other in G_2. Study the effects of these changes on the behavior of the Fiedler vector and the corresponding effect on your ability to correctly identify the two clustered subgraphs G_1 and G_2.

4.5. Select a network graph of interest to you. In the spirit of the case study in Section 4.4, identify a handful of questions, relating to the underlying complex system from which the network derives, for which the analysis of certain network graph characteristics may be relevant. Using software of your choice, conduct such an analysis and report on your findings.