CHAPTER 4 Graph Data

The traditional paradigm in data analysis typically assumes that each data instance is independent of another. However, often data instances may be connected or linked to other instances via various types of relationships. The instances themselves may be described by various attributes. What emerges is a network or graph of instances (or nodes), connected by links (or edges). Both the nodes and edges in the graph may have several attributes that may be numerical or categorical, or even more complex (e.g., time series data). Increasingly, today's massive data is in the form of such graphs or networks. Examples include the World Wide Web (with its Web pages and hyperlinks), social networks (wikis, blogs, tweets, and other social media data), semantic networks (ontologies), biological networks (protein interactions, gene regulation networks, metabolic pathways), citation networks for scientific literature, and so on. In this chapter we look at the analysis of the link structure in graphs that arise from these kinds of networks. We will study basic topological properties as well as models that give rise to such graphs.

4.1 GRAPH CONCEPTS

Graphs

Formally, a *graph* $G = (V, E)$ is a mathematical structure consisting of a finite nonempty set V of *vertices* or *nodes*, and a set $E \subseteq V \times V$ of *edges* consisting of *unordered* pairs of vertices. An edge from a node to itself, (v_i, v_i), is called a *loop*. An undirected graph without loops is called a *simple graph*. Unless mentioned explicitly, we will consider a graph to be simple. An edge $e = (v_i, v_j)$ between v_i and v_j is said to be *incident with* nodes v_i and v_j; in this case we also say that v_i and v_j are *adjacent* to one another, and that they are *neighbors*. The number of nodes in the graph G, given as $|V| = n$, is called the *order* of the graph, and the number of edges in the graph, given as $|E| = m$, is called the *size* of G.

A *directed graph* or *digraph* has an edge set E consisting of *ordered* pairs of vertices. A directed edge (v_i, v_j) is also called an *arc*, and is said to be *from* v_i *to* v_j. We also say that v_i is the *tail* and v_j the *head* of the arc.

A *weighted graph* consists of a graph together with a weight w_{ij} for each edge $(v_i, v_j) \in E$. Every graph can be considered to be a weighted graph in which the edges have weight one.

Subgraphs

A graph $H = (V_H, E_H)$ is called a *subgraph* of $G = (V, E)$ if $V_H \subseteq V$ and $E_H \subseteq E$. We also say that G is a *supergraph* of H. Given a subset of the vertices $V' \subseteq V$, the *induced subgraph* $G' = (V', E')$ consists exactly of all the edges present in G between vertices in V'. More formally, for all $v_i, v_j \in V'$, $(v_i, v_j) \in E' \iff (v_i, v_j) \in E$. In other words, two nodes are adjacent in G' if and only if they are adjacent in G. A (sub)graph is called *complete* (or a *clique*) if there exists an edge between all pairs of nodes.

Degree

The *degree* of a node $v_i \in V$ is the number of edges incident with it, and is denoted as $d(v_i)$ or just d_i. The *degree sequence* of a graph is the list of the degrees of the nodes sorted in non-increasing order.

Let N_k denote the number of vertices with degree k. The *degree frequency distribution* of a graph is given as

$$(N_0, N_1, \ldots, N_t)$$

where t is the maximum degree for a node in G. Let X be a random variable denoting the degree of a node. The *degree distribution* of a graph gives the probability mass function f for X, given as

$$\big(f(0), f(1), \ldots, f(t)\big)$$

where $f(k) = P(X = k) = \frac{N_k}{n}$ is the probability of a node with degree k, given as the number of nodes N_k with degree k, divided by the total number of nodes n. In graph analysis, we typically make the assumption that the input graph represents a population, and therefore we write f instead of $\hat{f}$ for the probability distributions.

For directed graphs, the *indegree* of node v_i, denoted as $id(v_i)$, is the number of edges with v_i as head, that is, the number of incoming edges at v_i. The *outdegree* of v_i, denoted $od(v_i)$, is the number of edges with v_i as the tail, that is, the number of outgoing edges from v_i.

Path and Distance

A *walk* in a graph G between nodes x and y is an ordered sequence of vertices, starting at x and ending at y,

$$x = v_0,\ v_1,\ \ldots,\ v_{t-1},\ v_t = y$$

such that there is an edge between every pair of consecutive vertices, that is, $(v_{i-1}, v_i) \in E$ for all $i = 1, 2, \ldots, t$. The length of the walk, t, is measured in terms of *hops* – the number of edges along the walk. In a walk, there is no restriction on the number of times a given vertex may appear in the sequence; thus both the vertices and edges may be repeated. A walk starting and ending at the same vertex (i.e., with $y = x$) is called *closed*. A *trail* is a walk with distinct edges, and a *path* is a walk with *distinct* vertices (with the exception of the start and end vertices). A closed path with length

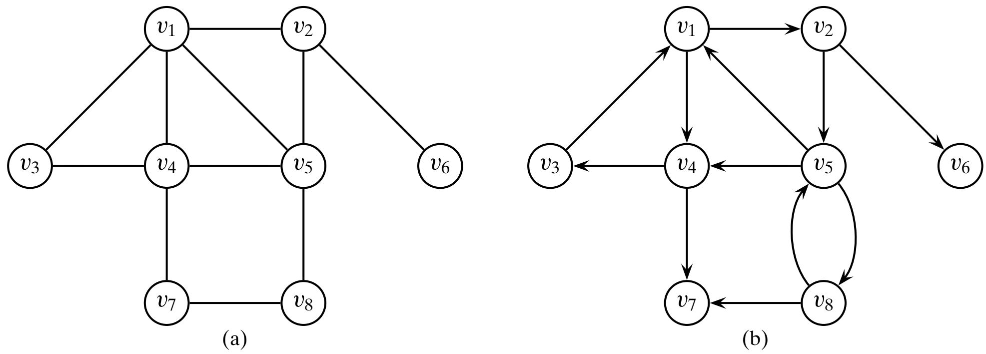


Figure 4.1. (a) A graph (undirected). (b) A directed graph.

$t \geq 3$ is called a *cycle*, that is, a cycle begins and ends at the same vertex and has distinct nodes.

A path of minimum length between nodes x and y is called a *shortest path*, and the length of the shortest path is called the *distance* between x and y, denoted as $d(x, y)$. If no path exists between the two nodes, the distance is assumed to be $d(x, y) = \infty$.

Connectedness

Two nodes v_i and v_j are said to be *connected* if there exists a path between them. A graph is *connected* if there is a path between all pairs of vertices. A *connected component*, or just *component*, of a graph is a maximal connected subgraph. If a graph has only one component it is connected; otherwise it is *disconnected*, as by definition there cannot be a path between two different components.

For a directed graph, we say that it is *strongly connected* if there is a (directed) path between all ordered pairs of vertices. We say that it is *weakly connected* if there exists a path between node pairs only by considering edges as undirected.

Example 4.1. Figure 4.1a shows a graph with $|V| = 8$ vertices and $|E| = 11$ edges. Because $(v_1, v_5) \in E$, we say that v_1 and v_5 are adjacent. The degree of v_1 is $d(v_1) = d_1 = 4$. The degree sequence of the graph is

$$(4, 4, 4, 3, 2, 2, 2, 1)$$

and therefore its degree frequency distribution is given as

$$(N_0, N_1, N_2, N_3, N_4) = (0, 1, 3, 1, 3)$$

We have $N_0 = 0$ because there are no isolated vertices, and $N_4 = 3$ because there are three nodes, v_1, v_4 and v_5, that have degree $k = 4$; the other numbers are obtained in a similar fashion. The degree distribution is given as

$$\big(f(0), f(1), f(2), f(3), f(4)\big) = (0, 0.125, 0.375, 0.125, 0.375)$$

The vertex sequence $(v_3, v_1, v_2, v_5, v_1, v_2, v_6)$ is a walk of length 6 between v_3 and v_6. We can see that vertices v_1 and v_2 have been visited more than once. In contrast, the vertex sequence $(v_3, v_4, v_7, v_8, v_5, v_2, v_6)$ is a path of length 6 between v_3 and v_6. However, this is not the shortest path between them, which happens to be (v_3, v_1, v_2, v_6) with length 3. Thus, the distance between them is given as $d(v_3, v_6) = 3$.

Figure 4.1b shows a directed graph with 8 vertices and 12 edges. We can see that edge (v_5, v_8) is distinct from edge (v_8, v_5). The indegree of v_7 is $id(v_7) = 2$, whereas its outdegree is $od(v_7) = 0$. Thus, there is no (directed) path from v_7 to any other vertex.

Adjacency Matrix

A graph $G = (V, E)$, with $|V| = n$ vertices, can be conveniently represented in the form of an $n \times n$, symmetric binary *adjacency matrix*, $\mathbf{A}$, defined as

$$\mathbf{A}(i, j) = \begin{cases} 1 & \text{if } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$

If the graph is directed, then the adjacency matrix $\mathbf{A}$ is not symmetric, as $(v_i, v_j) \in E$ obviously does not imply that $(v_j, v_i) \in E$.

If the graph is weighted, then we obtain an $n \times n$ *weighted adjacency matrix*, $\mathbf{A}$, defined as

$$\mathbf{A}(i, j) = \begin{cases} w_{ij} & \text{if } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$

where w_{ij} is the weight on edge $(v_i, v_j) \in E$. A weighted adjacency matrix can always be converted into a binary one, if desired, by using some threshold τ on the edge weights

$$\mathbf{A}(i, j) = \begin{cases} 1 & \text{if } w_{ij} \geq \tau \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

Graphs from Data Matrix

Many datasets that are not in the form of a graph can nevertheless be converted into one. Let $\mathbf{D} = \{\mathbf{x}_i\}_{i=1}^n$ (with $\mathbf{x}_i \in \mathbb{R}^d$), be a dataset consisting of n points in a d-dimensional space. We can define a weighted graph $G = (V, E)$, where there exists a node for each point in $\mathbf{D}$, and there exists an edge between each pair of points, with weight

$$w_{ij} = sim(\mathbf{x}_i, \mathbf{x}_j)$$

where $sim(\mathbf{x}_i, \mathbf{x}_j)$ denotes the similarity between points $\mathbf{x}_i$ and $\mathbf{x}_j$. For instance, similarity can be defined as being inversely related to the Euclidean distance between the points via the transformation

$$w_{ij} = sim(\mathbf{x}_i, \mathbf{x}_j) = \exp\left\{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right\} \tag{4.2}$$

where σ is the spread parameter (equivalent to the standard deviation in the normal density function). This transformation restricts the similarity function $sim()$ to lie in the range $[0, 1]$. One can then choose an appropriate threshold τ and convert the weighted adjacency matrix into a binary one via Eq. (4.1).

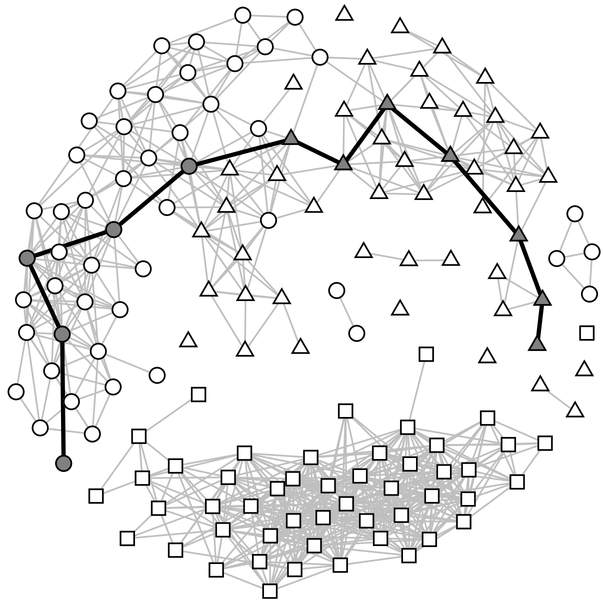


Figure 4.2. Iris similarity graph.

Example 4.2. Figure 4.2 shows the similarity graph for the Iris dataset (see Table 1.1). The pairwise similarity between distinct pairs of points was computed using Eq. (4.2), with $\sigma = 1/\sqrt{2}$ (we do not allow loops, to keep the graph simple). The mean similarity between points was 0.197, with a standard deviation of 0.290.

A binary adjacency matrix was obtained via Eq. (4.1) using a threshold of $\tau = 0.777$, which results in an edge between points having similarity higher than two standard deviations from the mean. The resulting Iris graph has 150 nodes and 753 edges.

The nodes in the Iris graph in Figure 4.2 have also been categorized according to their class. The circles correspond to class `iris-versicolor`, the triangles to `iris-virginica`, and the squares to `iris-setosa`. The graph has two big components, one of which is exclusively composed of nodes labeled as `iris-setosa`.

4.2 TOPOLOGICAL ATTRIBUTES

In this section we study some of the purely topological, that is, edge-based or structural, attributes of graphs. These attributes are *local* if they apply to only a single node (or an edge), and *global* if they refer to the entire graph.

Degree

We have already defined the degree of a node v_i as the number of its neighbors. A more general definition that holds even when the graph is weighted is as follows:

$$d_i = \sum_j \mathbf{A}(i, j)$$

The degree is clearly a local attribute of each node. One of the simplest global attribute is the *average degree*:

$$\mu_d = \frac{\sum_i d_i}{n}$$

The preceding definitions can easily be generalized for (weighted) directed graphs. For example, we can obtain the indegree and outdegree by taking the summation over the incoming and outgoing edges, as follows:

$$id(v_i) = \sum_j \mathbf{A}(j,i)$$

$$od(v_i) = \sum_j \mathbf{A}(i,j)$$

The average indegree and average outdegree can be obtained likewise.

Average Path Length

The *average path length*, also called the *characteristic path length*, of a connected graph is given as

$$\mu_L = \frac{\sum_i \sum_{j>i} d(v_i, v_j)}{\binom{n}{2}} = \frac{2}{n(n-1)} \sum_i \sum_{j>i} d(v_i, v_j)$$

where n is the number of nodes in the graph, and $d(v_i, v_j)$ is the distance between v_i and v_j. For a directed graph, the average is over all ordered pairs of vertices:

$$\mu_L = \frac{1}{n(n-1)} \sum_i \sum_j d(v_i, v_j)$$

For a disconnected graph the average is taken over only the connected pairs of vertices.

Eccentricity

The *eccentricity* of a node v_i is the maximum distance from v_i to any other node in the graph:

$$e(v_i) = \max_j \left\{ d(v_i, v_j) \right\}$$

If the graph is disconnected the eccentricity is computed only over pairs of vertices with finite distance, that is, only for vertices connected by a path.

Radius and Diameter

The *radius* of a connected graph, denoted $r(G)$, is the minimum eccentricity of any node in the graph:

$$r(G) = \min_i \left\{ e(v_i) \right\} = \min_i \left\{ \max_j \left\{ d(v_i, v_j) \right\} \right\}$$

The *diameter*, denoted $d(G)$, is the maximum eccentricity of any vertex in the graph:

$$d(G) = \max_i \{e(v_i)\} = \max_{i,j} \{d(v_i, v_j)\}$$

For a disconnected graph, the diameter is the maximum eccentricity over all the connected components of the graph.

The diameter of a graph G is sensitive to outliers. A more robust notion is *effective diameter*, defined as the minimum number of hops for which a large fraction, typically 90%, of all connected pairs of nodes can reach each other. More formally, let $H(k)$ denote the number of pairs of nodes that can reach each other in k hops or less. The effective diameter is defined as the smallest value of k such that $H(k) \geq 0.9 \times H(d(G))$.

Example 4.3. For the graph in Figure 4.1a, the eccentricity of node v_4 is $e(v_4) = 3$ because the node farthest from it is v_6 and $d(v_4, v_6) = 3$. The radius of the graph is $r(G) = 2$; both v_1 and v_5 have the least eccentricity value of 2. The diameter of the graph is $d(G) = 4$, as the largest distance over all the pairs is $d(v_6, v_7) = 4$.

The diameter of the Iris graph is $d(G) = 11$, which corresponds to the bold path connecting the gray nodes in Figure 4.2. The degree distribution for the Iris graph is shown in Figure 4.3. The numbers at the top of each bar indicate the frequency. For example, there are exactly 13 nodes with degree 7, which corresponds to the probability $f(7) = \frac{13}{150} = 0.0867$.

The path length histogram for the Iris graph is shown in Figure 4.4. For instance, 1044 node pairs have a distance of 2 hops between them. With $n = 150$ nodes, there

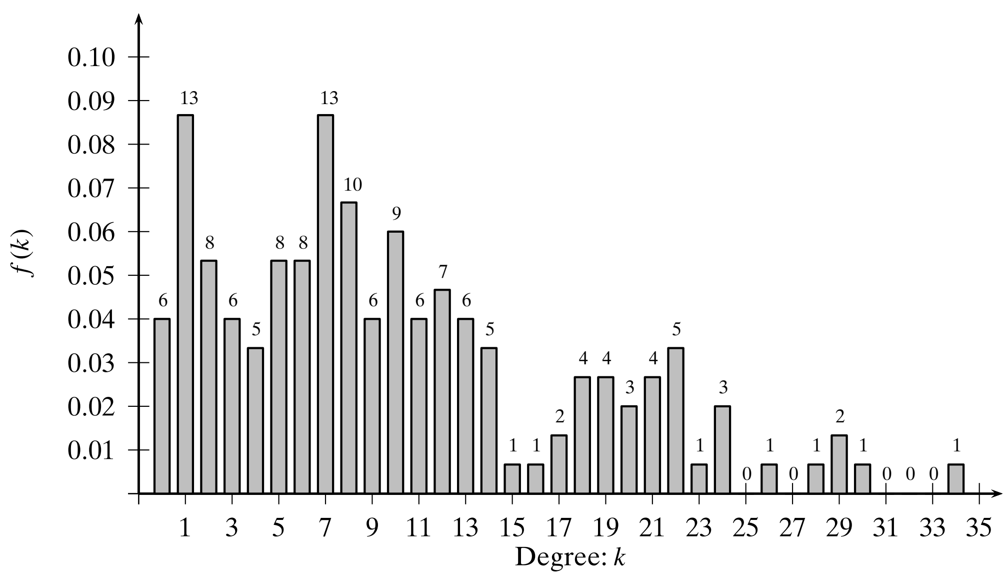


Figure 4.3. Iris graph: degree distribution.

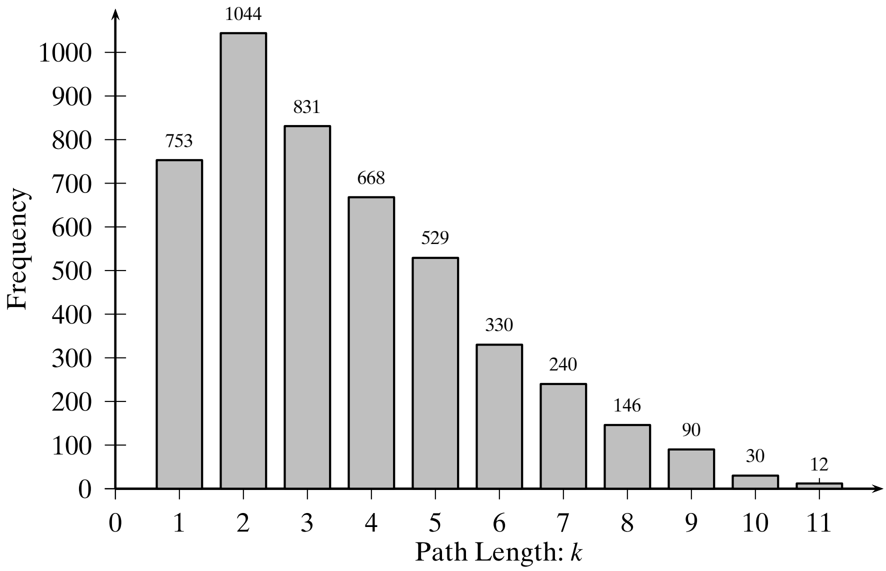


Figure 4.4. Iris graph: path length histogram.

are $\binom{n}{2} = 11,175$ pairs. Out of these 6502 pairs are unconnected, and there are a total of 4673 reachable pairs. Out of these $\frac{4175}{4673} = 0.89$ fraction are reachable in 6 hops, and $\frac{4415}{4673} = 0.94$ fraction are reachable in 7 hops. Thus, we can determine that the effective diameter is 7. The average path length is 3.58.

Clustering Coefficient

The *clustering coefficient* of a node v_i is a measure of the density of edges in the neighborhood of v_i. Let $G_i = (V_i, E_i)$ be the subgraph induced by the neighbors of vertex v_i. Note that $v_i \not\in V_i$, as we assume that G is simple. Let $|V_i| = n_i$ be the number of neighbors of v_i, and $|E_i| = m_i$ be the number of edges among the neighbors of v_i. The clustering coefficient of v_i is defined as

$$C(v_i) = \frac{\text{no. of edges in } G_i}{\text{maximum number of edges in } G_i} = \frac{m_i}{\binom{n_i}{2}} = \frac{2 \cdot m_i}{n_i(n_i - 1)}$$

The clustering coefficient gives an indication about the "cliquishness" of a node's neighborhood, because the denominator corresponds to the case when G_i is a complete subgraph.

The *clustering coefficient* of a graph G is simply the average clustering coefficient over all the nodes, given as

$$C(G) = \frac{1}{n} \sum_i C(v_i)$$

Because $C(v_i)$ is well defined only for nodes with degree $d(v_i) \geq 2$, we can define $C(v_i) = 0$ for nodes with degree less than 2. Alternatively, we can take the summation only over nodes with $d(v_i) \geq 2$.

The clustering coefficient $C(v_i)$ of a node is closely related to the notion of transitive relationships in a graph or network. That is, if there exists an edge between v_i and v_j, and another between v_i and v_k, then how likely are v_j and v_k to be linked or connected to each other. Define the subgraph composed of the edges (v_i, v_j) and (v_i, v_k) to be a *connected triple* centered at v_i. A connected triple centered at v_i that includes (v_j, v_k) is called a *triangle* (a complete subgraph of size 3). The clustering coefficient of node v_i can be expressed as

$$C(v_i) = \frac{\text{no. of triangles including } v_i}{\text{no. of connected triples centered at } v_i}$$

Note that the number of connected triples centered at v_i is simply $\binom{d_i}{2} = \frac{n_i(n_i-1)}{2}$, where $d_i = n_i$ is the number of neighbors of v_i.

Generalizing the aforementioned notion to the entire graph yields the *transitivity* of the graph, defined as

$$T(G) = \frac{3 \times \text{no. of triangles in } G}{\text{no. of connected triples in } G}$$

The factor 3 in the numerator is due to the fact that each triangle contributes to three connected triples centered at each of its three vertices. Informally, transitivity measures the degree to which a friend of your friend is also your friend, say, in a social network.

Efficiency

The *efficiency* for a pair of nodes v_i and v_j is defined as $\frac{1}{d(v_i,v_j)}$. If v_i and v_j are not connected, then $d(v_i, v_j) = \infty$ and the efficiency is $1/\infty = 0$. As such, the smaller the distance between the nodes, the more "efficient" the communication between them. The *efficiency* of a graph G is the average efficiency over all pairs of nodes, whether connected or not, given as

$$\frac{2}{n(n-1)} \sum_i \sum_{j>i} \frac{1}{d(v_i, v_j)}$$

The maximum efficiency value is 1, which holds for a complete graph.

The *local efficiency* for a node v_i is defined as the efficiency of the subgraph G_i induced by the neighbors of v_i. Because $v_i \notin G_i$, the local efficiency is an indication of the local fault tolerance, that is, how efficient is the communication between neighbors of v_i when v_i is removed or deleted from the graph.

Example 4.4. For the graph in Figure 4.1a, consider node v_4. Its neighborhood graph is shown in Figure 4.5. The clustering coefficient of node v_4 is given as

$$C(v_4) = \frac{2}{\binom{4}{2}} = \frac{2}{6} = 0.33$$

The clustering coefficient for the entire graph (over all nodes) is given as

$$C(G) = \frac{1}{8}\left(\frac{1}{2} + \frac{1}{3} + 1 + \frac{1}{3} + \frac{1}{3} + 0 + 0 + 0\right) = \frac{2.5}{8} = 0.3125$$

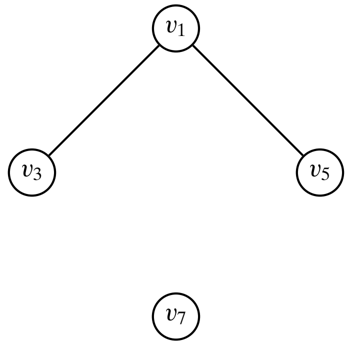


Figure 4.5. Subgraph G_4 induced by node v_4.

The local efficiency of v_4 is given as

$$\frac{2}{4 \cdot 3}\left(\frac{1}{d(v_1, v_3)} + \frac{1}{d(v_1, v_5)} + \frac{1}{d(v_1, v_7)} + \frac{1}{d(v_3, v_5)} + \frac{1}{d(v_3, v_7)} + \frac{1}{d(v_5, v_7)}\right)$$
$$= \frac{1}{6}(1 + 1 + 0 + 0.5 + 0 + 0) = \frac{2.5}{6} = 0.417$$

4.3 CENTRALITY ANALYSIS

The notion of *centrality* is used to rank the vertices of a graph in terms of how "central" or important they are. A centrality can be formally defined as a function $c\colon V \to \mathbb{R}$, that induces a total order on V. We say that v_i is at least as central as v_j if $c(v_i) \geq c(v_j)$.

4.3.1 Basic Centralities

Degree Centrality

The simplest notion of centrality is the degree d_i of a vertex v_i – the higher the degree, the more important or central the vertex. For directed graphs, one may further consider the indegree centrality and outdegree centrality of a vertex.

Eccentricity Centrality

According to this notion, the less eccentric a node is, the more central it is. Eccentricity centrality is thus defined as follows:

$$c(v_i) = \frac{1}{e(v_i)} = \frac{1}{\max_j \left\{d(v_i, v_j)\right\}}$$

A node v_i that has the least eccentricity, that is, for which the eccentricity equals the graph radius, $e(v_i) = r(G)$, is called a *center node*, whereas a node that has the highest eccentricity, that is, for which eccentricity equals the graph diameter, $e(v_i) = d(G)$, is called a *periphery node*.

Eccentricity centrality is related to the problem of *facility location*, that is, choosing the optimum location for a resource or facility. The central node minimizes the maximum distance to any node in the network, and thus the most central node would be an ideal location for, say, a hospital, because it is desirable to minimize the maximum distance someone has to travel to get to the hospital quickly.

Closeness Centrality

Whereas eccentricity centrality uses the maximum of the distances from a given node, closeness centrality uses the sum of all the distances to rank how central a node is

$$c(v_i) = \frac{1}{\sum_j d(v_i, v_j)}$$

A node v_i with the smallest total distance, $\sum_j d(v_i, v_j)$, is called the *median node*.

Closeness centrality optimizes a different objective function for the facility location problem. It tries to minimize the total distance over all the other nodes, and thus a median node, which has the highest closeness centrality, is the optimal one to, say, locate a facility such as a new coffee shop or a mall, as in this case it is not as important to minimize the distance for the farthest node.

Betweenness Centrality

For a given vertex v_i the betweenness centrality measures how many shortest paths between all pairs of vertices include v_i. This gives an indication as to the central "monitoring" role played by v_i for various pairs of nodes. Let η_{jk} denote the number of shortest paths between vertices v_j and v_k, and let $\eta_{jk}(v_i)$ denote the number of such paths that include or contain v_i. Then the fraction of paths through v_i is denoted as

$$\gamma_{jk}(v_i) = \frac{\eta_{jk}(v_i)}{\eta_{jk}}$$

If the two vertices v_j and v_k are not connected, we assume $\gamma_{jk} = 0$.

The betweenness centrality for a node v_i is defined as

$$c(v_i) = \sum_{j \neq i} \sum_{\substack{k \neq i \\ k > j}} \gamma_{jk} = \sum_{j \neq i} \sum_{\substack{k \neq i \\ k > j}} \frac{\eta_{jk}(v_i)}{\eta_{jk}} \tag{4.3}$$

Example 4.5. Consider Figure 4.1a. The values for the different node centrality measures are given in Table 4.1. According to degree centrality, nodes v_1, v_4, and v_5 are the most central. The eccentricity centrality is the highest for the center nodes in the graph, which are v_1 and v_5. It is the least for the periphery nodes, of which there are two, v_6 and, v_7.

Nodes v_1 and v_5 have the highest closeness centrality value. In terms of betweenness, vertex v_5 is the most central, with a value of 6.5. We can compute this value by considering only those pairs of nodes v_j and v_k that have at least one shortest

Table 4.1. Centrality values

Centrality	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
Degree	4	3	2	4	4	1	2	2
Eccentricity	0.5	0.33	0.33	0.33	0.5	0.25	0.25	0.33
$e(v_i)$	2	3	3	3	2	4	4	3
Closeness	0.100	0.083	0.071	0.091	0.100	0.056	0.067	0.071
$\sum_j d(v_i, v_j)$	10	12	14	11	10	18	15	14
Betweenness	4.5	6	0	5	6.5	0	0.83	1.17

path passing through v_5, as only these node pairs have $\gamma_{jk} > 0$ in Eq. (4.3). We have

$$\begin{aligned} c(v_5) &= \gamma_{18} + \gamma_{24} + \gamma_{27} + \gamma_{28} + \gamma_{38} + \gamma_{46} + \gamma_{48} + \gamma_{67} + \gamma_{68} \\ &= 1 + \frac{1}{2} + \frac{2}{3} + 1 + \frac{2}{3} + \frac{1}{2} + \frac{1}{2} + \frac{2}{3} + 1 = 6.5 \end{aligned}$$

4.3.2 Web Centralities

We now consider directed graphs, especially in the context of the Web. For example, hypertext documents have directed links pointing from one document to another; citation networks of scientific articles have directed edges from a paper to the cited papers, and so on. We consider notions of centrality that are particularly suited to such Web-scale graphs.

Prestige

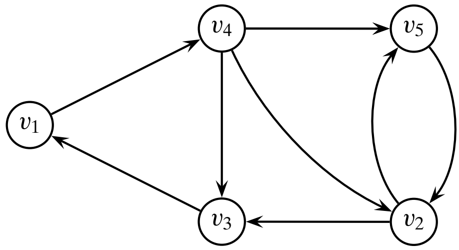
We first look at the notion of *prestige*, or the *eigenvector centrality*, of a node in a directed graph. As a centrality, prestige is supposed to be a measure of the importance or rank of a node. Intuitively the more the links that point to a given node, the higher its prestige. However, prestige does not depend simply on the indegree; it also (recursively) depends on the prestige of the nodes that point to it.

Let $G = (V, E)$ be a directed graph, with $|V| = n$. The adjacency matrix of G is an $n \times n$ asymmetric matrix $\mathbf{A}$ given as

$$\mathbf{A}(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E \\ 0 & \text{if } (u, v) \notin E \end{cases}$$

Let $p(u)$ be a positive real number, called the *prestige* score for node u. Using the intuition that the prestige of a node depends on the prestige of other nodes pointing to it, we can obtain the prestige score of a given node v as follows:

$$\begin{aligned} p(v) &= \sum_u \mathbf{A}(u, v) \cdot p(u) \\ &= \sum_u \mathbf{A}^T(v, u) \cdot p(u) \end{aligned}$$



(a)

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

(b)

$$\mathbf{A}^T = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

(c)

Figure 4.6. Example graph (a), adjacency matrix (b), and its transpose (c).

For example, in Figure 4.6, the prestige of v_5 depends on the prestige of v_2 and v_4.

Across all the nodes, we can recursively express the prestige scores as

$$\mathbf{p}' = \mathbf{A}^T\mathbf{p} \tag{4.4}$$

where $\mathbf{p}$ is an n-dimensional column vector corresponding to the prestige scores for each vertex.

Starting from an initial prestige vector we can use Eq. (4.4) to obtain an updated prestige vector in an iterative manner. In other words, if $\mathbf{p}_{k-1}$ is the prestige vector across all the nodes at iteration $k-1$, then the updated prestige vector at iteration k is given as

$$\begin{aligned}
\mathbf{p}_k &= \mathbf{A}^T\mathbf{p}_{k-1} \\
&= \mathbf{A}^T(\mathbf{A}^T\mathbf{p}_{k-2}) = \left(\mathbf{A}^T\right)^2 \mathbf{p}_{k-2} \\
&= \left(\mathbf{A}^T\right)^2 (\mathbf{A}^T\mathbf{p}_{k-3}) = \left(\mathbf{A}^T\right)^3 \mathbf{p}_{k-3} \\
&= \vdots \\
&= \left(\mathbf{A}^T\right)^k \mathbf{p}_0
\end{aligned}$$

where $\mathbf{p}_0$ is the initial prestige vector. It is well known that the vector $\mathbf{p}_k$ converges to the dominant eigenvector of $\mathbf{A}^T$ with increasing k.

The dominant eigenvector of $\mathbf{A}^T$ and the corresponding eigenvalue can be computed using the *power iteration* approach whose pseudo-code is shown in Algorithm 4.1. The method starts with the vector $\mathbf{p}_0$, which can be initialized to the vector $(1, 1, \ldots, 1)^T \in \mathbb{R}^n$. In each iteration, we multiply on the left by $\mathbf{A}^T$, and scale the intermediate $\mathbf{p}_k$ vector by dividing it by the maximum entry $\mathbf{p}_k[i]$ in $\mathbf{p}_k$ to prevent numeric overflow. The ratio of the maximum entry in iteration k to that in $k-1$, given as $\lambda = \frac{\mathbf{p}_k[i]}{\mathbf{p}_{k-1}[i]}$, yields an estimate for the eigenvalue. The iterations continue until the difference between successive eigenvector estimates falls below some threshold $\epsilon > 0$.

ALGORITHM 4.1. Power Iteration Method: Dominant Eigenvector

POWERITERATION $(\mathbf{A}, \epsilon)$:

$k \leftarrow 0$ `// iteration`
$\mathbf{p}_0 \leftarrow \mathbf{1} \in \mathbb{R}^n$ `// initial vector`
repeat
  $k \leftarrow k+1$
  $\mathbf{p}_k \leftarrow \mathbf{A}^T \mathbf{p}_{k-1}$ `// eigenvector estimate`
  $i \leftarrow \arg\max_j \{\mathbf{p}_k[j]\}$ `// maximum value index`
  $\lambda \leftarrow \mathbf{p}_k[i]/\mathbf{p}_{k-1}[i]$ `// eigenvalue estimate`
  $\mathbf{p}_k \leftarrow \frac{1}{\mathbf{p}_k[i]}\mathbf{p}_k$ `// scale vector`
until $\|\mathbf{p}_k - \mathbf{p}_{k-1}\| \le \epsilon$
$\mathbf{p} \leftarrow \frac{1}{\|\mathbf{p}_k\|}\mathbf{p}_k$ `// normalize eigenvector`
return $\mathbf{p}, \lambda$

Table 4.2. Power method via scaling

$\mathbf{p}_0$	$\mathbf{p}_1$	$\mathbf{p}_2$	$\mathbf{p}_3$
$\begin{pmatrix}1\\1\\1\\1\\1\end{pmatrix}$	$\begin{pmatrix}1\\2\\2\\1\\2\end{pmatrix} \rightarrow \begin{pmatrix}0.5\\1\\1\\0.5\\1\end{pmatrix}$	$\begin{pmatrix}1\\1.5\\1.5\\0.5\\1.5\end{pmatrix} \rightarrow \begin{pmatrix}0.67\\1\\1\\0.33\\1\end{pmatrix}$	$\begin{pmatrix}1\\1.33\\1.33\\0.67\\1.33\end{pmatrix} \rightarrow \begin{pmatrix}0.75\\1\\1\\0.5\\1\end{pmatrix}$
λ	2	1.5	1.33

$\mathbf{p}_4$	$\mathbf{p}_5$	$\mathbf{p}_6$	$\mathbf{p}_7$
$\begin{pmatrix}1\\1.5\\1.5\\0.75\\1.5\end{pmatrix} \rightarrow \begin{pmatrix}0.67\\1\\1\\0.5\\1\end{pmatrix}$	$\begin{pmatrix}1\\1.5\\1.5\\0.67\\1.5\end{pmatrix} \rightarrow \begin{pmatrix}0.67\\1\\1\\0.44\\1\end{pmatrix}$	$\begin{pmatrix}1\\1.44\\1.44\\0.67\\1.44\end{pmatrix} \rightarrow \begin{pmatrix}0.69\\1\\1\\0.46\\1\end{pmatrix}$	$\begin{pmatrix}1\\1.46\\1.46\\0.69\\1.46\end{pmatrix} \rightarrow \begin{pmatrix}0.68\\1\\1\\0.47\\1\end{pmatrix}$
1.5	1.5	1.444	1.462

Example 4.6. Consider the example shown in Figure 4.6. Starting with an initial prestige vector $\mathbf{p}_0 = (1,1,1,1,1)^T$, in Table 4.2 we show several iterations of the power method for computing the dominant eigenvector of $\mathbf{A}^T$. In each iteration we obtain $\mathbf{p}_k = \mathbf{A}^T\mathbf{p}_{k-1}$. For example,

$$\mathbf{p}_1 = \mathbf{A}^T\mathbf{p}_0 = \begin{pmatrix}0&0&1&0&0\\0&0&0&1&1\\0&1&0&1&0\\1&0&0&0&0\\0&1&0&1&0\end{pmatrix}\begin{pmatrix}1\\1\\1\\1\\1\end{pmatrix} = \begin{pmatrix}1\\2\\2\\1\\2\end{pmatrix}$$

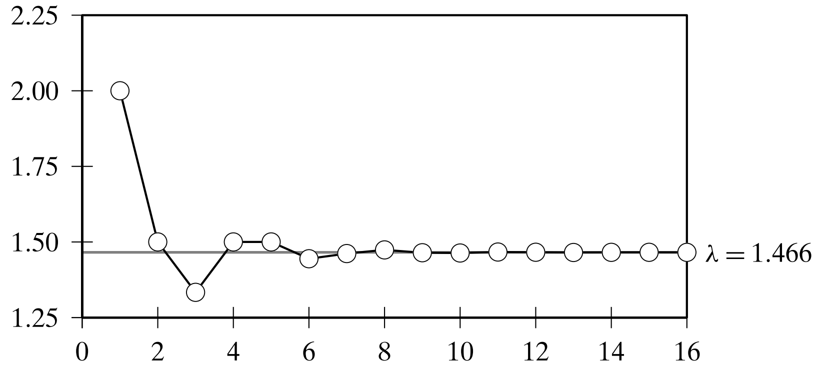


Figure 4.7. Convergence of the ratio to dominant eigenvalue.

Before the next iteration, we scale $\mathbf{p}_1$ by dividing each entry by the maximum value in the vector, which is 2 in this case, to obtain

$$\mathbf{p}_1 = \frac{1}{2}\begin{pmatrix}1\\1\\2\\1\\2\end{pmatrix} = \begin{pmatrix}0.5\\1\\1\\0.5\\1\end{pmatrix}$$

As k becomes large, we get

$$\mathbf{p}_k = \mathbf{A}^T\mathbf{p}_{k-1} \simeq \lambda\mathbf{p}_{k-1}$$

which implies that the ratio of the maximum element of $\mathbf{p}_k$ to that of $\mathbf{p}_{k-1}$ should approach λ. The table shows this ratio for successive iterations. We can see in Figure 4.7 that within 10 iterations the ratio converges to $\lambda = 1.466$. The scaled dominant eigenvector converges to

$$\mathbf{p}_k = \begin{pmatrix}1\\1.466\\1.466\\0.682\\1.466\end{pmatrix}$$

After normalizing it to be a unit vector, the dominant eigenvector is given as

$$\mathbf{p} = \begin{pmatrix}0.356\\0.521\\0.521\\0.243\\0.521\end{pmatrix}$$

Thus, in terms of prestige, v_2, v_3, and v_5 have the highest values, as all of them have indegree 2 and are pointed to by nodes with the same incoming values of prestige. On the other hand, although v_1 and v_4 have the same indegree, v_1 is ranked higher, because v_3 contributes its prestige to v_1, but v_4 gets its prestige only from v_1.

PageRank

PageRank is a method for computing the prestige or centrality of nodes in the context of Web search. The Web graph consists of pages (the nodes) connected by hyperlinks (the edges). The method uses the so-called *random surfing* assumption that a person surfing the Web randomly chooses one of the outgoing links from the current page, or with some very small probability randomly jumps to any of the other pages in the Web graph. The PageRank of a Web page is defined to be the probability of a random web surfer landing at that page. Like prestige, the PageRank of a node v recursively depends on the PageRank of other nodes that point to it.

Normalized Prestige We assume for the moment that each node u has outdegree at least 1. We discuss later how to handle the case when a node has no outgoing edges. Let $od(u) = \sum_v \mathbf{A}(u,v)$ denote the outdegree of node u. Because a random surfer can choose among any of its outgoing links, if there is a link from u to v, then the probability of visiting v from u is $\frac{1}{od(u)}$.

Starting from an initial probability or PageRank $p_0(u)$ for each node, such that

$$\sum_u p_0(u) = 1$$

we can compute an updated PageRank vector for v as follows:

$$\begin{aligned} p(v) &= \sum_u \frac{\mathbf{A}(u,v)}{od(u)} \cdot p(u) \\ &= \sum_u \mathbf{N}(u,v) \cdot p(u) \\ &= \sum_u \mathbf{N}^T(v,u) \cdot p(u) \end{aligned} \tag{4.5}$$

where $\mathbf{N}$ is the normalized adjacency matrix of the graph, given as

$$\mathbf{N}(u,v) = \begin{cases} \frac{1}{od(u)} & \text{if } (u,v) \in E \\ 0 & \text{if } (u,v) \notin E \end{cases}$$

Across all nodes, we can express the PageRank vector as follows:

$$\mathbf{p}' = \mathbf{N}^T \mathbf{p} \tag{4.6}$$

So far, the PageRank vector is essentially a normalized prestige vector.

Random Jumps In the random surfing approach, there is a small probability of jumping from one node to any of the other nodes in the graph, even if they do not have a link between them. In essence, one can think of the Web graph as a (virtual) fully connected directed graph, with an adjacency matrix given as

$$\mathbf{A}_r = \mathbf{1}_{n \times n} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

Here $\mathbf{1}_{n\times n}$ is the $n \times n$ matrix of all ones. For the random surfer matrix, the outdegree of each node is $od(u) = n$, and the probability of jumping from u to any node v is simply $\frac{1}{od(u)} = \frac{1}{n}$. Thus, if one allows only random jumps from one node to another, the PageRank can be computed analogously to Eq. (4.5):

$$
\begin{aligned}
p(v) &= \sum_u \frac{\mathbf{A}_r(u,v)}{od(u)} \cdot p(u) \\
&= \sum_u \mathbf{N}_r(u,v) \cdot p(u) \\
&= \sum_u \mathbf{N}_r^T(v,u) \cdot p(u)
\end{aligned}
$$

where $\mathbf{N}_r$ is the normalized adjacency matrix of the fully connected Web graph, given as

$$
\mathbf{N}_r = \begin{pmatrix} \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \end{pmatrix} = \frac{1}{n}\mathbf{A}_r = \frac{1}{n}\mathbf{1}_{n\times n}
$$

Across all the nodes the random jump PageRank vector can be represented as

$$
\mathbf{p}' = \mathbf{N}_r^T \mathbf{p}
$$

PageRank The full PageRank is computed by assuming that with some small probability, α, a random Web surfer jumps from the current node u to any other random node v, and with probability $1-\alpha$ the user follows an existing link from u to v. In other words, we combine the normalized prestige vector, and the random jump vector, to obtain the final PageRank vector, as follows:

$$
\begin{aligned}
\mathbf{p}' &= (1-\alpha)\mathbf{N}^T\mathbf{p} + \alpha\mathbf{N}_r^T\mathbf{p} \\
&= \big((1-\alpha)\mathbf{N}^T + \alpha\mathbf{N}_r^T\big)\mathbf{p} \\
&= \mathbf{M}^T\mathbf{p}
\end{aligned} \tag{4.7}
$$

where $\mathbf{M} = (1-\alpha)\mathbf{N} + \alpha\mathbf{N}_r$ is the combined normalized adjacency matrix. The PageRank vector can be computed in an iterative manner, starting with an initial PageRank assignment $\mathbf{p}_0$, and updating it in each iteration using Eq. (4.7). One minor problem arises if a node u does not have any outgoing edges, that is, when $od(u) = 0$. Such a node acts like a sink for the normalized prestige score. Because there is no outgoing edge from u, the only choice u has is to simply jump to another random node. Thus, we need to make sure that if $od(u) = 0$ then for the row corresponding to u in $\mathbf{M}$, denoted as $\mathbf{M}_u$, we set $\alpha = 1$, that is,

$$
\mathbf{M}_u = \begin{cases} \mathbf{M}_u & \text{if } od(u) > 0 \\ \frac{1}{n}\mathbf{1}_n^T & \text{if } od(u) = 0 \end{cases}
$$

where $\mathbf{1}_n$ is the n-dimensional vector of all ones. We can use the power iteration method in Algorithm 4.1 to compute the dominant eigenvector of $\mathbf{M}^T$.

Example 4.7. Consider the graph in Figure 4.6. The normalized adjacency matrix is given as

$$\mathbf{N} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0.33 & 0.33 & 0 & 0.33 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Because there are $n = 5$ nodes in the graph, the normalized random jump adjacency matrix is given as

$$\mathbf{N}_r = \begin{pmatrix} 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \end{pmatrix}$$

Assuming that $\alpha = 0.1$, the combined normalized adjacency matrix is given as

$$\mathbf{M} = 0.9\mathbf{N} + 0.1\mathbf{N}_r = \begin{pmatrix} 0.02 & 0.02 & 0.02 & 0.92 & 0.02 \\ 0.02 & 0.02 & 0.47 & 0.02 & 0.47 \\ 0.92 & 0.02 & 0.02 & 0.02 & 0.02 \\ 0.02 & 0.32 & 0.32 & 0.02 & 0.32 \\ 0.02 & 0.92 & 0.02 & 0.02 & 0.02 \end{pmatrix}$$

Computing the dominant eigenvector and eigenvalue of $\mathbf{M}^T$ we obtain $\lambda = 1$ and

$$\mathbf{p} = \begin{pmatrix} 0.419 \\ 0.546 \\ 0.417 \\ 0.422 \\ 0.417 \end{pmatrix}$$

Node v_2 has the highest PageRank value.

Hub and Authority Scores

Note that the PageRank of a node is independent of any query that a user may pose, as it is a global value for a Web page. However, for a specific user query, a page with a high global PageRank may not be that relevant. One would like to have a query-specific notion of the PageRank or prestige of a page. The Hyperlink Induced Topic Search (HITS) method is designed to do this. In fact, it computes two values to judge the importance of a page. The *authority score* of a page is analogous to PageRank or prestige, and it depends on how many "good" pages point to it. On the other hand, the *hub score* of a page is based on how many "good" pages it points to. In other words, a page with high authority has many hub pages pointing to it, and a page with high hub score points to many pages that have high authority.

Given a user query the HITS method first uses standard search engines to retrieve the set of relevant pages. It then expands this set to include any pages that point to some page in the set, or any pages that are pointed to by some page in the set. Any pages originating from the same host are eliminated. HITS is applied only on this expanded query specific graph G.

We denote by $a(u)$ the authority score and by $h(u)$ the hub score of node u. The authority score depends on the hub score and vice versa in the following manner:

$$a(v) = \sum_u \mathbf{A}^T(v,u) \cdot h(u)$$

$$h(v) = \sum_u \mathbf{A}(v,u) \cdot a(u)$$

In matrix notation, we obtain

$$\mathbf{a}' = \mathbf{A}^T\mathbf{h}$$

$$\mathbf{h}' = \mathbf{A}\mathbf{a}$$

In fact, we can rewrite the above recursively as follows:

$$\mathbf{a}_k = \mathbf{A}^T\mathbf{h}_{k-1} = \mathbf{A}^T(\mathbf{A}\mathbf{a}_{k-1}) = (\mathbf{A}^T\mathbf{A})\mathbf{a}_{k-1}$$

$$\mathbf{h}_k = \mathbf{A}\mathbf{a}_{k-1} = \mathbf{A}(\mathbf{A}^T\mathbf{h}_{k-1}) = (\mathbf{A}\mathbf{A}^T)\mathbf{h}_{k-1}$$

In other words, as $k \to \infty$, the authority score converges to the dominant eigenvector of $\mathbf{A}^T\mathbf{A}$, whereas the hub score converges to the dominant eigenvector of $\mathbf{A}\mathbf{A}^T$. The power iteration method can be used to compute the eigenvector in both cases. Starting with an initial authority vector $\mathbf{a} = \mathbf{1}_n$, the vector of all ones, we can compute the vector $\mathbf{h} = \mathbf{A}\mathbf{a}$. To prevent numeric overflows, we scale the vector by dividing by the maximum element. Next, we can compute $\mathbf{a} = \mathbf{A}^T\mathbf{h}$, and scale it too, which completes one iteration. This process is repeated until both $\mathbf{a}$ and $\mathbf{h}$ converge.

Example 4.8. For the graph in Figure 4.6, we can iteratively compute the authority and hub score vectors, by starting with $\mathbf{a} = (1,1,1,1,1)^T$. In the first iteration, we have

$$\mathbf{h} = \mathbf{A}\mathbf{a} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 1 \\ 3 \\ 1 \end{pmatrix}$$

After scaling by dividing by the maximum value 3, we get

$$\mathbf{h}' = \begin{pmatrix} 0.33 \\ 0.67 \\ 0.33 \\ 1 \\ 0.33 \end{pmatrix}$$

Next we update $\mathbf{a}$ as follows:

$$\mathbf{a} = \mathbf{A}^T\mathbf{h}' = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0.33 \\ 0.67 \\ 0.33 \\ 1 \\ 0.33 \end{pmatrix} = \begin{pmatrix} 0.33 \\ 1.33 \\ 1.67 \\ 0.33 \\ 1.67 \end{pmatrix}$$

After scaling by dividing by the maximum value 1.67, we get

$$\mathbf{a}' = \begin{pmatrix} 0.2 \\ 0.8 \\ 1 \\ 0.2 \\ 1 \end{pmatrix}$$

This sets the stage for the next iteration. The process continues until $\mathbf{a}$ and $\mathbf{h}$ converge to the dominant eigenvectors of $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}\mathbf{A}^T$, respectively, given as

$$\mathbf{a} = \begin{pmatrix} 0 \\ 0.46 \\ 0.63 \\ 0 \\ 0.63 \end{pmatrix} \qquad \mathbf{h} = \begin{pmatrix} 0 \\ 0.58 \\ 0 \\ 0.79 \\ 0.21 \end{pmatrix}$$

From these scores, we conclude that v_4 has the highest hub score because it points to three nodes – v_2, v_3, and v_5 – with good authority. On the other hand, both v_3 and v_5 have high authority scores, as the two nodes v_4 and v_2 with the highest hub scores point to them.

4.4 GRAPH MODELS

Surprisingly, many real-world networks exhibit certain common characteristics, even though the underlying data can come from vastly different domains, such as social networks, biological networks, telecommunication networks, and so on. A natural question is to understand the underlying processes that might give rise to such real-world networks. We consider several network measures that will allow us to compare and contrast different graph models. Real-world networks are usually *large* and *sparse*. By large we mean that the order or the number of nodes n is very large, and by sparse we mean that the graph size or number of edges $m = O(n)$. The models we study below make a similar assumption that the graphs are large and sparse.

Small-world Property

It has been observed that many real-world graphs exhibit the so-called *small-world* property that there is a short path between any pair of nodes. We say that a graph G exhibits small-world behavior if the average path length μ_L scales logarithmically with the number of nodes in the graph, that is, if

$$\mu_L \propto \log n$$

where n is the number of nodes in the graph. A graph is said to have *ultra-small-world* property if the average path length is much smaller than $\log n$, that is, if $\mu_L \ll \log n$.

Scale-free Property

In many real-world graphs it has been observed that the empirical degree distribution $f(k)$ exhibits a *scale-free* behavior captured by a power-law relationship with k, that is, the probability that a node has degree k satisfies the condition

$$f(k) \propto k^{-\gamma} \tag{4.8}$$

Intuitively, a power law indicates that the vast majority of nodes have very small degrees, whereas there are a few "hub" nodes that have high degrees, that is, they connect to or interact with lots of nodes. A power-law relationship leads to a scale-free or scale invariant behavior because scaling the argument by some constant c does not change the proportionality. To see this, let us rewrite Eq. (4.8) as an equality by introducing a proportionality constant α that does not depend on k, that is,

$$f(k) = \alpha k^{-\gamma} \tag{4.9}$$

Then we have

$$f(ck) = \alpha(ck)^{-\gamma} = (\alpha c^{-\gamma})k^{-\gamma} \propto k^{-\gamma}$$

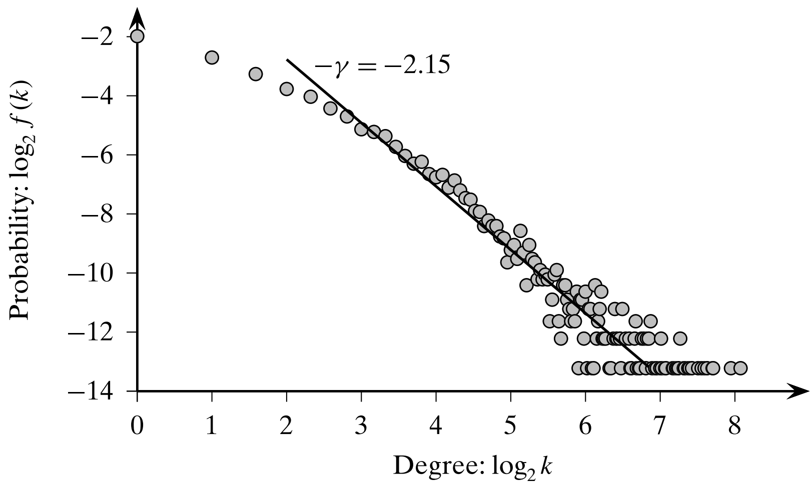
Also, taking the logarithm on both sides of Eq. (4.9) gives

$$\log f(k) = \log(\alpha k^{-\gamma})$$
$$\text{or } \log f(k) = -\gamma \log k + \log \alpha$$

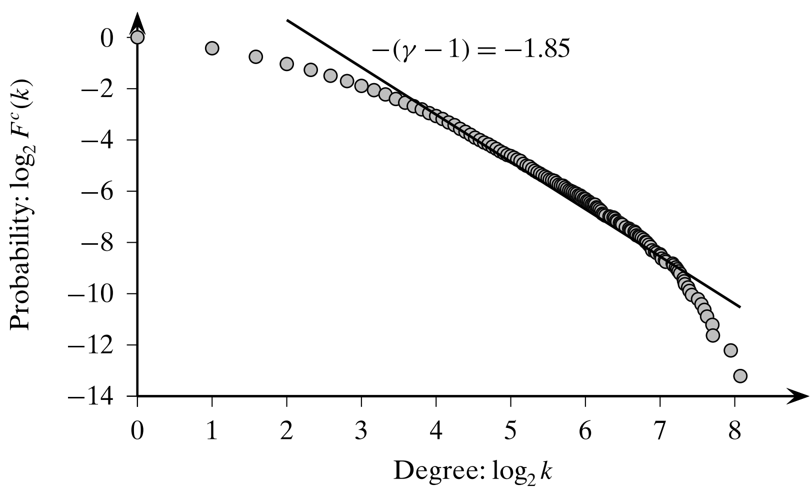
which is the equation of a straight line in the log-log plot of k versus $f(k)$, with $-\gamma$ giving the slope of the line. Thus, the usual approach to check whether a graph has scale-free behavior is to perform a least-square fit of the points $\big(\log k, \log f(k)\big)$ to a line, as illustrated in Figure 4.8a.

In practice, one of the problems with estimating the degree distribution for a graph is the high level of noise for the higher degrees, where frequency counts are the lowest. One approach to address the problem is to use the cumulative degree distribution $F(k)$, which tends to smooth out the noise. In particular, we use $F^c(k) = 1 - F(k)$, which gives the probability that a randomly chosen node has degree greater than k. If $f(k) \propto k^{-\gamma}$, and assuming that $\gamma > 1$, we have

$$\begin{aligned}
F^c(k) &= 1 - F(k) = 1 - \sum_{0}^{k} f(x) = \sum_{k}^{\infty} f(x) = \sum_{k}^{\infty} x^{-\gamma} \\
&\simeq \int_k^{\infty} x^{-\gamma} dx = \frac{x^{-\gamma+1}}{-\gamma+1}\bigg|_k^{\infty} = \frac{1}{(\gamma-1)} \cdot k^{-(\gamma-1)} \\
&\propto k^{-(\gamma-1)}
\end{aligned}$$



(a) Degree distribution



(b) Cumulative degree distribution

Figure 4.8. Degree distribution and its cumulative distribution.

In other words, the log-log plot of $F^c(k)$ versus k will also be a power law with slope $-(\gamma - 1)$ as opposed to $-\gamma$. Owing to the smoothing effect, plotting $\log k$ versus $\log F^c(k)$ and observing the slope gives a better estimate of the power law, as illustrated in Figure 4.8b.

Clustering Effect

Real-world graphs often also exhibit a *clustering effect*, that is, two nodes are more likely to be connected if they share a common neighbor. The clustering effect is captured by a high clustering coefficient for the graph G. Let $C(k)$ denote the average clustering coefficient for all nodes with degree k; then the clustering effect also manifests itself as a power-law relationship between $C(k)$ and k:

$$C(k) \propto k^{-\gamma}$$

In other words, a log-log plot of k versus $C(k)$ exhibits a straight line behavior with negative slope $-\gamma$. Intuitively, the power-law behavior indicates hierarchical clustering of the nodes. That is, nodes that are sparsely connected (i.e., have smaller degrees) are part of highly clustered areas (i.e., have higher average clustering coefficients). Further, only a few hub nodes (with high degrees) connect these clustered areas (the hub nodes have smaller clustering coefficients).

Example 4.9. Figure 4.8a plots the degree distribution for a graph of human protein interactions, where each node is a protein and each edge indicates if the two incident proteins interact experimentally. The graph has $n = 9521$ nodes and $m = 37,060$ edges. A linear relationship between $\log k$ and $\log f(k)$ is clearly visible, although very small and very large degree values do not fit the linear trend. The best fit line after ignoring the extremal degrees yields a value of $\gamma = 2.15$. The plot of $\log k$ versus $\log F^c(k)$ makes the linear fit quite prominent. The slope obtained here is $-(\gamma - 1) = 1.85$, that is, $\gamma = 2.85$. We can conclude that the graph exhibits scale-free behavior (except at the degree extremes), with γ somewhere between 2 and 3, as is typical of many real-world graphs.

The diameter of the graph is $d(G) = 14$, which is very close to $\log_2 n = \log_2(9521) = 13.22$. The network is thus small-world.

Figure 4.9 plots the average clustering coefficient as a function of degree. The log-log plot has a very weak linear trend, as observed from the line of best fit that gives a slope of $-\gamma = -0.55$. We can conclude that the graph exhibits weak hierarchical clustering behavior.

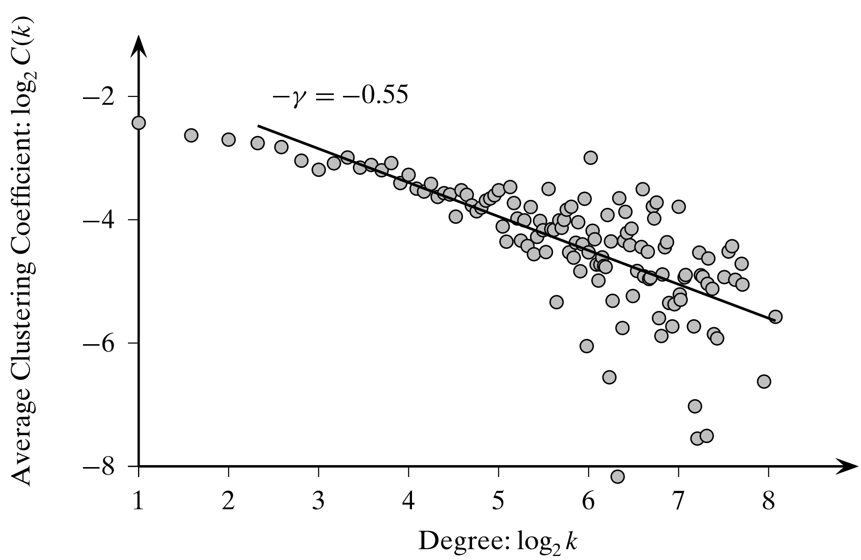


Figure 4.9. Average clustering coefficient distribution.

4.4.1 Erdös–Rényi Random Graph Model

The Erdös–Rényi (ER) model generates a random graph such that any of the possible graphs with a fixed number of nodes and edges has equal probability of being chosen.

The ER model has two parameters: the number of nodes n and the number of edges m. Let M denote the maximum number of edges possible among the n nodes, that is,

$$M = \binom{n}{2} = \frac{n(n-1)}{2}$$

The ER model specifies a collection of graphs $\mathcal{G}(n,m)$ with n nodes and m edges, such that each graph $G \in \mathcal{G}$ has equal probability of being selected:

$$P(G) = \frac{1}{\binom{M}{m}} = \binom{M}{m}^{-1}$$

where $\binom{M}{m}$ is the number of possible graphs with m edges (with n nodes) corresponding to the ways of choosing the m edges out of a total of M possible edges.

Let $V = \{v_1, v_2, \dots, v_n\}$ denote the set of n nodes. The ER method chooses a random graph $G = (V,E) \in \mathcal{G}$ via a generative process. At each step, it randomly selects two distinct vertices $v_i, v_j \in V$, and adds an edge (v_i, v_j) to E, provided the edge is not already in the graph G. The process is repeated until exactly m edges have been added to the graph.

Let X be a random variable denoting the degree of a node for $G \in \mathcal{G}$. Let p denote the probability of an edge in G, which can be computed as

$$p = \frac{m}{M} = \frac{m}{\binom{n}{2}} = \frac{2m}{n(n-1)}$$

Average Degree

For any given node in G its degree can be at most $n-1$ (because we do not allow loops). Because p is the probability of an edge for any node, the random variable X, corresponding to the degree of a node, follows a binomial distribution with probability of success p, given as

$$f(k) = P(X=k) = \binom{n-1}{k} p^k (1-p)^{n-1-k}$$

The average degree μ_d is then given as the expected value of X:

$$\mu_d = E[X] = (n-1)p$$

We can also compute the variance of the degrees among the nodes by computing the variance of X:

$$\sigma_d^2 = var(X) = (n-1)p(1-p)$$

Degree Distribution

To obtain the degree distribution for large and sparse random graphs, we need to derive an expression for $f(k) = P(X=k)$ as $n \to \infty$. Assuming that $m = O(n)$, we can write $p = \frac{m}{n(n-1)/2} = \frac{O(n)}{n(n-1)/2} = \frac{1}{O(n)} \to 0$. In other words, we are interested in the asymptotic behavior of the graphs as $n \to \infty$ and $p \to 0$.

Under these two trends, notice that the expected value and variance of X can be rewritten as

$$E[X] = (n-1)p \simeq np \text{ as } n \to \infty$$
$$var(X) = (n-1)p(1-p) \simeq np \text{ as } n \to \infty \text{ and } p \to 0$$

In other words, for large and sparse random graphs the expectation and variance of X are the same:

$$E[X] = var(X) = np$$

and the binomial distribution can be approximated by a Poisson distribution with parameter λ, given as

$$f(k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

where $\lambda = np$ represents both the expected value and variance of the distribution. Using Stirling's approximation of the factorial $k! \simeq k^k e^{-k}\sqrt{2\pi k}$ we obtain

$$f(k) = \frac{\lambda^k e^{-\lambda}}{k!} \simeq \frac{\lambda^k e^{-\lambda}}{k^k e^{-k}\sqrt{2\pi k}} = \frac{e^{-\lambda}}{\sqrt{2\pi}} \frac{(\lambda e)^k}{\sqrt{k}k^k}$$

In other words, we have

$$f(k) \propto \alpha^k k^{-\frac{1}{2}} k^{-k}$$

for $\alpha = \lambda e = npe$. We conclude that large and sparse random graphs follow a Poisson degree distribution, which does not exhibit a power-law relationship. Thus, in one crucial respect, the ER random graph model is not adequate to describe real-world scale-free graphs.

Clustering Coefficient

Let us consider a node v_i in G with degree k. The clustering coefficient of v_i is given as

$$C(v_i) = \frac{2m_i}{k(k-1)}$$

where $k = n_i$ also denotes the number of nodes and m_i denotes the number of edges in the subgraph induced by neighbors of v_i. However, because p is the probability of an edge, the expected number of edges m_i among the neighbors of v_i is simply

$$m_i = \frac{pk(k-1)}{2}$$

Thus, we obtain

$$C(v_i) = \frac{2m_i}{k(k-1)} = p$$

In other words, the expected clustering coefficient across all nodes of all degrees is uniform, and thus the overall clustering coefficient is also uniform:

$$C(G) = \frac{1}{n}\sum_i C(v_i) = p$$

Furthermore, for sparse graphs we have $p \to 0$, which in turn implies that $C(G) = C(v_i) \to 0$. Thus, large random graphs have no clustering effect whatsoever, which is contrary to many real-world networks.

Diameter

We saw earlier that the expected degree of a node is $\mu_d = \lambda$, which means that within one hop from a given node, we can reach λ other nodes. Because each of the neighbors of the initial node also has average degree λ, we can approximate the number of nodes that are two hops away as λ^2. In general, at a coarse level of approximation (i.e., ignoring shared neighbors), we can estimate the number of nodes at a distance of k hops away from a starting node v_i as λ^k. However, because there are a total of n distinct vertices in the graph, we have

$$\sum_{k=1}^{t} \lambda^k = n$$

where t denotes the maximum number of hops from v_i. We have

$$\sum_{k=1}^{t} \lambda^k = \frac{\lambda^{t+1}-1}{\lambda-1} \simeq \lambda^t$$

Plugging into the expression above, we have

$$\lambda^t \simeq n \quad \text{or}$$
$$t \log\lambda \simeq \log n \quad \text{which implies}$$
$$t \simeq \frac{\log n}{\log\lambda} \propto \log n$$

Because the path length from a node to the farthest node is bounded by t, it follows that the diameter of the graph is also bounded by that value, that is,

$$d(G) \propto \log n$$

assuming that the expected degree λ is fixed. We can thus conclude that random graphs satisfy at least one property of real-world graphs, namely that they exhibit small-world behavior.

4.4.2 Watts–Strogatz Small-world Graph Model

The random graph model fails to exhibit a high clustering coefficient, but it is small-world. The Watts–Strogatz (WS) model tries to explicitly model high local clustering by starting with a regular network in which each node is connected to its k neighbors on the right and left, assuming that the initial n vertices are arranged in a large circular backbone. Such a network will have a high clustering coefficient, but will not be small-world. Surprisingly, adding a small amount of randomness in the regular network by randomly *rewiring* some of the edges or by adding a small fraction of random edges leads to the emergence of the small-world phenomena.

The WS model starts with n nodes arranged in a circular layout, with each node connected to its immediate left and right neighbors. The edges in the initial layout are

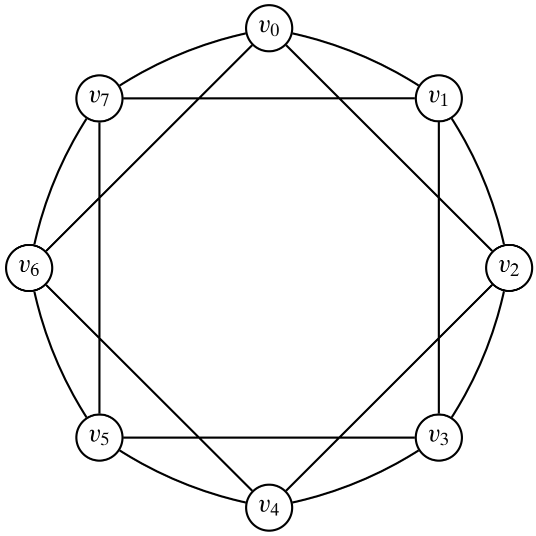


Figure 4.10. Watts–Strogatz regular graph: $n = 8$, $k = 2$.

called *backbone* edges. Each node has edges to an additional $k - 1$ neighbors to the left and right. Thus, the WS model starts with a *regular* graph of degree $2k$, where each node is connected to its k neighbors on the right and k neighbors on the left, as illustrated in Figure 4.10.

Clustering Coefficient and Diameter of Regular Graph

Consider the subgraph G_v induced by the $2k$ neighbors of a node v. The clustering coefficient of v is given as

$$C(v) = \frac{m_v}{M_v} \tag{4.10}$$

where m_v is the actual number of edges, and M_v is the maximum possible number of edges, among the neighbors of v.

To compute m_v, consider some node r_i that is at a distance of i hops (with $1 \leq i \leq k$) from v to the right, considering only the backbone edges. The node r_i has edges to $k - i$ of its immediate right neighbors (restricted to the right neighbors of v), and to $k - 1$ of its left neighbors (all k left neighbors, excluding v). Owing to the symmetry about v, a node l_i that is at a distance of i backbone hops from v to the left has the same number of edges. Thus, the degree of any node in G_v that is i backbone hops away from v is given as

$$d_i = (k - i) + (k - 1) = 2k - i - 1$$

Because each edge contributes to the degree of its two incident nodes, summing the degrees of all neighbors of v, we obtain

$$2m_v = 2\left(\sum_{i=1}^{k} 2k - i - 1\right)$$

$$m_v = 2k^2 - \frac{k(k+1)}{2} - k$$
$$m_v = \frac{3}{2}k(k-1) \tag{4.11}$$

On the other hand, the number of possible edges among the $2k$ neighbors of v is given as

$$M_v = \binom{2k}{2} = \frac{2k(2k-1)}{2} = k(2k-1)$$

Plugging the expressions for m_v and M_v into Eq. (4.10), the clustering coefficient of a node v is given as

$$C(v) = \frac{m_v}{M_v} = \frac{3k-3}{4k-2}$$

As k increases, the clustering coefficient approaches $\frac{3}{4}$ because $C(G) = C(v) \to \frac{3}{4}$ as $k \to \infty$.

The WS regular graph thus has a high clustering coefficient. However, it does not satisfy the small-world property. To see this, note that along the backbone, the farthest node from v has a distance of at most $\frac{n}{2}$ hops. Further, because each node is connected to k neighbors on either side, one can reach the farthest node in at most $\frac{n/2}{k}$ hops. More precisely, the diameter of a regular WS graph is given as

$$d(G) = \begin{cases} \lceil \frac{n}{2k} \rceil & \text{if } n \text{ is even} \\ \lceil \frac{n-1}{2k} \rceil & \text{if } n \text{ is odd} \end{cases}$$

The regular graph has a diameter that scales linearly in the number of nodes, and thus it is not small-world.

Random Perturbation of Regular Graph

Edge Rewiring Starting with the regular graph of degree $2k$, the WS model perturbs the regular structure by adding some randomness to the network. One approach is to randomly rewire edges with probability r. That is, for each edge (u, v) in the graph, with probability r, replace v with another randomly chosen node avoiding loops and duplicate edges. Because the WS regular graph has $m = kn$ total edges, after rewiring, rm of the edges are random, and $(1-r)m$ are regular.

Edge Shortcuts An alternative approach is that instead of rewiring edges, we add a few *shortcut* edges between random pairs of nodes, as shown in Figure 4.11. The total number of random shortcut edges added to the network is given as $mr = knr$, so that r can be considered as the probability, per edge, of adding a shortcut edge. The total number of edges in the graph is then simply $m + mr = (1+r)m = (1+r)kn$. Because $r \in [0, 1]$, the number of edges then lies in the range $[kn, 2kn]$.

In either approach, if the probability r of rewiring or adding shortcut edges is $r = 0$, then we are left with the original regular graph, with high clustering coefficient, but with no small-world property. On the other hand, if the rewiring or shortcut probability $r = 1$, the regular structure is disrupted, and the graph approaches a random graph, with little to no clustering effect, but with small-world property. Surprisingly, introducing

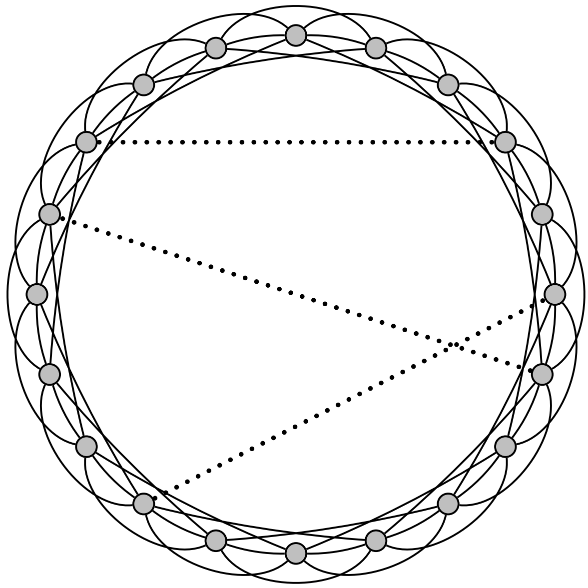


Figure 4.11. Watts–Strogatz graph ($n = 20$, $k = 3$): shortcut edges are shown dotted.

only a small amount of randomness leads to a significant change in the regular network. As one can see in Figure 4.11, the presence of a few long-range shortcuts reduces the diameter of the network significantly. That is, even for a low value of r, the WS model retains most of the regular local clustering structure, but at the same time becomes small-world.

Properties of Watts–Strogatz Graphs

Degree Distribution Let us consider the shortcut approach, which is easier to analyze. In this approach, each vertex has degree at least $2k$. In addition there are the shortcut edges, which follow a binomial distribution. Each node can have $n' = n - 2k - 1$ additional shortcut edges, so we take n' as the number of independent trials to add edges. Because a node has degree $2k$, with shortcut edge probability of r, we expect roughly $2kr$ shortcuts from that node, but the node can connect to at most $n - 2k - 1$ other nodes. Thus, we can take the probability of success as

$$p = \frac{2kr}{n-2k-1} = \frac{2kr}{n'} \tag{4.12}$$

Let X denote the random variable denoting the number of shortcuts for each node. Then the probability of a node with j shortcut edges is given as

$$f(j) = P(X = j) = \binom{n'}{j} p^j (1-p)^{n'-j}$$

with $E[X] = n'p = 2kr$. The expected degree of each node in the network is therefore

$$2k + E[X] = 2k + 2kr = 2k(1+r)$$

It is clear that the degree distribution of the WS graph does not adhere to a power law. Thus, such networks are not scale-free.

Clustering Coefficient After the shortcut edges have been added, each node v has expected degree $2k(1+r)$, that is, it is on average connected to $2kr$ new neighbors, in addition to the $2k$ original ones. The number of possible edges among v's neighbors is given as

$$M_v = \frac{2k(1+r)(2k(1+r)-1)}{2} = (1+r)k(4kr+2k-1)$$

Because the regular WS graph remains intact even after adding shortcuts, the neighbors of v retain all $\frac{3k(k-1)}{2}$ initial edges, as given in Eq. (4.11). In addition, some of the shortcut edges may link pairs of nodes among v's neighbors. Let Y be the random variable that denotes the number of shortcut edges present among the $2k(1+r)$ neighbors of v; then Y follows a binomial distribution with probability of success p, as given in Eq. (4.12). Thus, the expected number of shortcut edges is given as

$$E[Y] = pM_v$$

Let m_v be the random variable corresponding to the actual number of edges present among v's neighbors, whether regular or shortcut edges. The expected number of edges among the neighbors of v is then given as

$$E[m_v] = E\left[\frac{3k(k-1)}{2} + Y\right] = \frac{3k(k-1)}{2} + pM_v$$

Because the binomial distribution is essentially concentrated around the mean, we can now approximate the clustering coefficient by using the expected number of edges, as follows:

$$\begin{aligned} C(v) &\simeq \frac{E[m_v]}{M_v} = \frac{\frac{3k(k-1)}{2} + pM_v}{M_v} = \frac{3k(k-1)}{2M_v} + p \\ &= \frac{3(k-1)}{(1+r)(4kr+2(2k-1))} + \frac{2kr}{n-2k-1} \end{aligned}$$

using the value of p given in Eq. (4.12). For large graphs we have $n \to \infty$, so we can drop the second term above, to obtain

$$C(v) \simeq \frac{3(k-1)}{(1+r)(4kr+2(2k-1))} = \frac{3k-3}{4k-2+2r(2kr+4k-1)} \tag{4.13}$$

As $r \to 0$, the above expression becomes equivalent to Eq. (4.10). Thus, for small values of r the clustering coefficient remains high.

Diameter Deriving an analytical expression for the diameter of the WS model with random edge shortcuts is not easy. Instead we resort to an empirical study of the behavior of WS graphs when a small number of random shortcuts are added. In Example 4.10 we find that small values of shortcut edge probability r are enough to reduce the diameter from $O(n)$ to $O(\log n)$. The WS model thus leads to graphs that are small-world and that also exhibit the clustering effect. However, the WS graphs do not display a scale-free degree distribution.

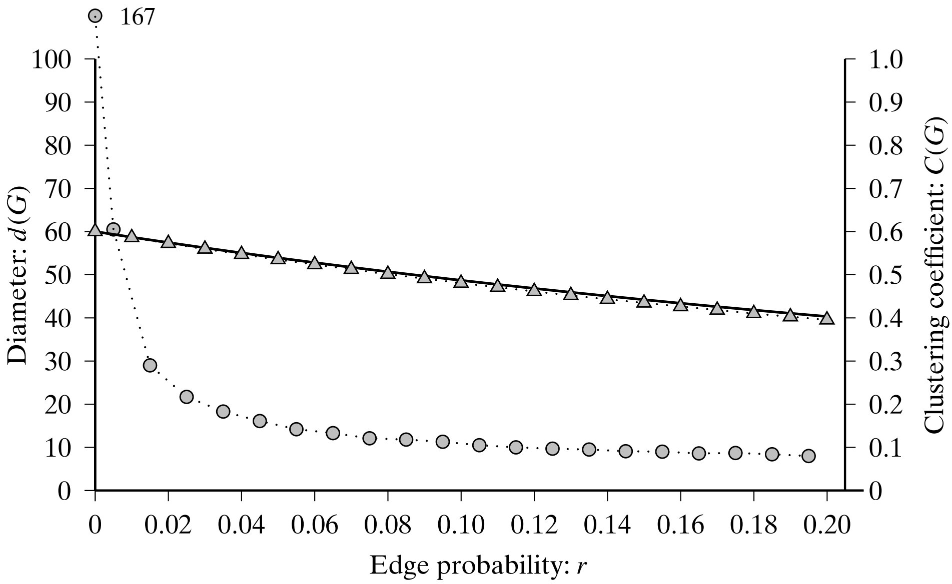


Figure 4.12. Watts-Strogatz model: diameter (circles) and clustering coefficient (triangles).

Example 4.10. Figure 4.12 shows a simulation of the WS model, for a graph with $n = 1000$ vertices and $k = 3$. The x-axis shows different values of the probability r of adding random shortcut edges. The diameter values are shown as circles using the left y-axis, whereas the clustering values are shown as triangles using the right y-axis. These values are the averages over 10 runs of the WS model. The solid line gives the clustering coefficient from the analytical formula in Eq. (4.13), which is in perfect agreement with the simulation values.

The initial regular graph has diameter

$$d(G) = \left\lceil \frac{n}{2k} \right\rceil = \left\lceil \frac{1000}{6} \right\rceil = 167$$

and its clustering coefficient is given as

$$C(G) = \frac{3(k-1)}{2(2k-1)} = \frac{6}{10} = 0.6$$

We can observe that the diameter quickly reduces, even with very small edge addition probability. For $r = 0.005$, the diameter is 61. For $r = 0.1$, the diameter shrinks to 11, which is on the same scale as $O(\log_2 n)$ because $\log_2 1000 \simeq 10$. On the other hand, we can observe that clustering coefficient remains high. For $r = 0.1$, the clustering coefficient is 0.48. Thus, the simulation study confirms that the addition of even a small number of random shortcut edges reduces the diameter of the WS regular graph from $O(n)$ (large-world) to $O(\log n)$ (small-world). At the same time the graph retains its local clustering property.

4.4.3 Barabási–Albert Scale-free Model

The Barabási–Albert (BA) model tries to capture the scale-free degree distributions of real-world graphs via a generative process that adds new nodes and edges at each time step. Further, the edge growth is based on the concept of *preferential attachment*; that is, edges from the new vertex are more likely to link to nodes with higher degrees. For this reason the model is also known as the *rich get richer* approach. The BA model mimics a dynamically growing graph by adding new vertices and edges at each time-step $t = 1, 2, \ldots$. Let G_t denote the graph at time t, and let n_t denote the number of nodes, and m_t the number of edges in G_t.

Initialization

The BA model starts at time-step $t = 0$, with an initial graph G_0 with n_0 nodes and m_0 edges. Each node in G_0 should have degree at least 1; otherwise it will never be chosen for preferential attachment. We will assume that each node has initial degree 2, being connected to its left and right neighbors in a circular layout. Thus $m_0 = n_0$.

Growth and Preferential Attachment

The BA model derives a new graph G_{t+1} from G_t by adding exactly one new node u and adding $q \le n_0$ new edges from u to q distinct nodes $v_j \in G_t$, where node v_j is chosen with probability $\pi_t(v_j)$ proportional to its degree in G_t, given as

$$\pi_t(v_j) = \frac{d_j}{\sum_{v_i \in G_t} d_i} \tag{4.14}$$

Because only one new vertex is added at each step, the number of nodes in G_t is given as

$$n_t = n_0 + t$$

Further, because exactly q new edges are added at each time-step, the number of edges in G_t is given as

$$m_t = m_0 + qt$$

Because the sum of the degrees is two times the number of edges in the graph, we have

$$\sum_{v_i \in G_t} d(v_i) = 2m_t = 2(m_0 + qt)$$

We can thus rewrite Eq. (4.14) as

$$\pi_t(v_j) = \frac{d_j}{2(m_0 + qt)} \tag{4.15}$$

As the network grows, owing to preferential attachment, one intuitively expects high degree hubs to emerge.

Example 4.11. Figure 4.13 shows a graph generated according to the BA model, with parameters $n_0 = 3, q = 2$, and $t = 12$. Initially, at time $t = 0$, the graph has $n_0 = 3$ vertices, namely $\{v_0, v_1, v_2\}$ (shown in gray), connected by $m_0 = 3$ edges (shown in bold). At each time step $t = 1, \ldots, 12$, vertex v_{t+2} is added to the growing network

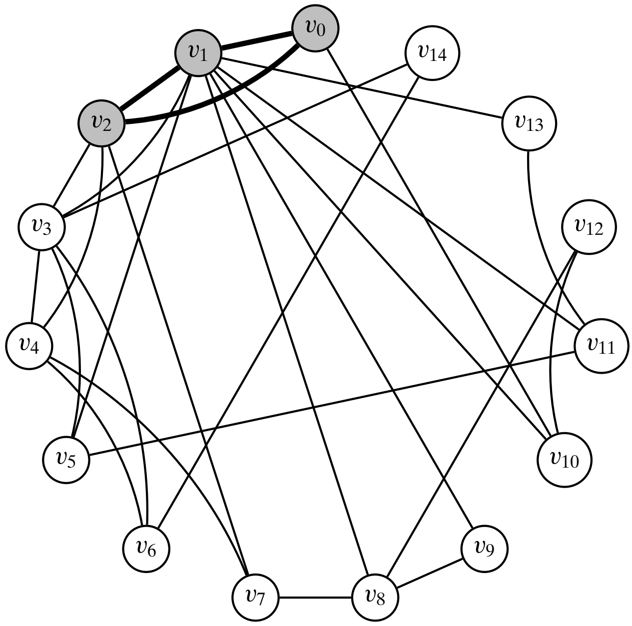


Figure 4.13. Barabási–Albert graph ($n_0 = 3$, $q = 2$, $t = 12$).

and is connected to $q = 2$ vertices chosen with a probability proportional to their degree.

For example, at $t = 1$, vertex v_3 is added, with edges to v_1 and v_2, chosen according to the distribution

$$\pi_0(v_i) = 1/3 \text{ for } i = 0, 1, 2$$

At $t = 2$, v_4 is added. Using Eq. (4.15), nodes v_2 and v_3 are preferentially chosen according to the probability distribution

$$\pi_1(v_0) = \pi_1(v_3) = \frac{2}{10} = 0.2$$
$$\pi_1(v_1) = \pi_1(v_2) = \frac{3}{10} = 0.3$$

The final graph after $t = 12$ time-steps shows the emergence of some hub nodes, such as v_1 (with degree 9) and v_3 (with degree 6).

Degree Distribution

We now study two different approaches to estimate the degree distribution for the BA model, namely the discrete approach, and the continuous approach.

Discrete Approach The discrete approach is also called the *master-equation* method. Let X_t be a random variable denoting the degree of a node in G_t, and let $f_t(k)$ denote the probability mass function for X_t. That is, $f_t(k)$ is the degree distribution for the graph G_t at time-step t. Simply put, $f_t(k)$ is the fraction of nodes with degree k at time t. Let n_t denote the number of nodes and m_t the number of edges in G_t. Further, let $n_t(k)$ denote the number of nodes with degree k in G_t. Then we have

$$f_t(k) = \frac{n_t(k)}{n_t}$$

Because we are interested in large real-world graphs, as $t \to \infty$, the number of nodes and edges in G_t can be approximated as

$$\begin{aligned} n_t &= n_0 + t \simeq t \\ m_t &= m_0 + qt \simeq qt \end{aligned} \tag{4.16}$$

Based on Eq. (4.14), at time-step $t+1$, the probability $\pi_t(k)$ that some node with degree k in G_t is chosen for preferential attachment can be written as

$$\pi_t(k) = \frac{k \cdot n_t(k)}{\sum_i i \cdot n_t(i)}$$

Dividing the numerator and denominator by n_t, we have

$$\pi_t(k) = \frac{k \cdot \frac{n_t(k)}{n_t}}{\sum_i i \cdot \frac{n_t(i)}{n_t}} = \frac{k \cdot f_t(k)}{\sum_i i \cdot f_t(i)} \tag{4.17}$$

Note that the denominator is simply the expected value of X_t, that is, the mean degree in G_t, because

$$E[X_t] = \mu_d(G_t) = \sum_i i \cdot f_t(i) \tag{4.18}$$

Note also that in any graph the average degree is given as

$$\mu_d(G_t) = \frac{\sum_i d_i}{n_t} = \frac{2m_t}{n_t} \simeq \frac{2qt}{t} = 2q \tag{4.19}$$

where we used Eq. (4.16), that is, $m_t = qt$. Equating Eqs. (4.18) and (4.19), we can rewrite the preferential attachment probability [Eq. (4.17)] for a node of degree k as

$$\pi_t(k) = \frac{k \cdot f_t(k)}{2q} \tag{4.20}$$

We now consider the change in the number of nodes with degree k, when a new vertex u joins the growing network at time-step $t+1$. The net change in the number of nodes with degree k is given as the number of nodes with degree k at time $t+1$ minus the number of nodes with degree k at time t, given as

$$(n_t + 1) \cdot f_{t+1}(k) - n_t \cdot f_t(k)$$

Using the approximation that $n_t \simeq t$ from Eq. (4.16), the net change in degree k nodes is

$$(n_t + 1) \cdot f_{t+1}(k) - n_t \cdot f_t(k) = (t+1) \cdot f_{t+1}(k) - t \cdot f_t(k) \tag{4.21}$$

The number of nodes with degree k increases whenever u connects to a vertex v_i of degree $k-1$ in G_t, as in this case v_i will have degree k in G_{t+1}. Over the q edges added at time $t+1$, the number of nodes with degree $k-1$ in G_t that are chosen to connect to u is given as

$$q\pi_t(k-1) = \frac{q \cdot (k-1) \cdot f_t(k-1)}{2q} = \frac{1}{2} \cdot (k-1) \cdot f_t(k-1) \tag{4.22}$$

where we use Eq. (4.20) for $\pi_t(k-1)$. Note that Eq. (4.22) holds only when $k > q$. This is because v_i must have degree at least q, as each node that is added at time $t \geq 1$ has initial degree q. Therefore, if $d_i = k-1$, then $k-1 \geq q$ implies that $k > q$ (we can also ensure that the initial n_0 edges have degree q by starting with clique of size $n_0 = q+1$).

At the same time, the number of nodes with degree k decreases whenever u connects to a vertex v_i with degree k in G_t, as in this case v_i will have a degree $k+1$ in G_{t+1}. Using Eq. (4.20), over the q edges added at time $t+1$, the number of nodes with degree k in G_t that are chosen to connect to u is given as

$$q \cdot \pi_t(k) = \frac{q \cdot k \cdot f_t(k)}{2q} = \frac{1}{2} \cdot k \cdot f_t(k) \tag{4.23}$$

Based on the preceding discussion, when $k > q$, the net change in the number of nodes with degree k is given as the difference between Eqs. (4.22) and (4.23) in G_t:

$$q \cdot \pi_t(k-1) - q \cdot \pi_t(k) = \frac{1}{2} \cdot (k-1) \cdot f_t(k-1) - \frac{1}{2} k \cdot f_t(k) \tag{4.24}$$

Equating Eqs. (4.21) and (4.24) we obtain the master equation for $k > q$:

$$(t+1) \cdot f_{t+1}(k) - t \cdot f_t(k) = \frac{1}{2} \cdot (k-1) \cdot f_t(k-1) - \frac{1}{2} \cdot k \cdot f_t(k) \tag{4.25}$$

On the other hand, when $k = q$, assuming that there are no nodes in the graph with degree less than q, then only the newly added node contributes to an increase in the number of nodes with degree $k = q$ by one. However, if u connects to an existing node v_i with degree k, then there will be a decrease in the number of degree k nodes because in this case v_i will have degree $k+1$ in G_{t+1}. The net change in the number of nodes with degree k is therefore given as

$$1 - q \cdot \pi_t(k) = 1 - \frac{1}{2} \cdot k \cdot f_t(k) \tag{4.26}$$

Equating Eqs. (4.21) and (4.26) we obtain the master equation for the boundary condition $k = q$:

$$(t+1) \cdot f_{t+1}(k) - t \cdot f_t(k) = 1 - \frac{1}{2} \cdot k \cdot f_t(k) \tag{4.27}$$

Our goal is now to obtain the stationary or time-invariant solutions for the master equations. In other words, we study the solution when

$$f_{t+1}(k) = f_t(k) = f(k) \tag{4.28}$$

The stationary solution gives the degree distribution that is independent of time.

Let us first derive the stationary solution for $k = q$. Substituting Eq. (4.28) into Eq. (4.27) and setting $k = q$, we obtain

$$(t+1)\cdot f(q) - t\cdot f(q) = 1 - \frac{1}{2}\cdot q\cdot f(q)$$
$$2f(q) = 2 - q\cdot f(q), \text{ which implies that}$$
$$f(q) = \frac{2}{q+2} \tag{4.29}$$

The stationary solution for $k > q$ gives us a recursion for $f(k)$ in terms of $f(k-1)$:

$$(t+1)\cdot f(k) - t\cdot f(k) = \frac{1}{2}\cdot (k-1)\cdot f(k-1) - \frac{1}{2}\cdot k\cdot f(k)$$
$$2f(k) = (k-1)\cdot f(k-1) - k\cdot f(k), \text{ which implies that}$$
$$f(k) = \left(\frac{k-1}{k+2}\right)\cdot f(k-1) \tag{4.30}$$

Expanding (4.30) until the boundary condition $k = q$ yields

$$\begin{aligned}
f(k) &= \frac{(k-1)}{(k+2)}\cdot f(k-1)\\
&= \frac{(k-1)(k-2)}{(k+2)(k+1)}\cdot f(k-2)\\
&\quad\vdots\\
&= \frac{(k-1)(k-2)(k-3)(k-4)\cdots(q+3)(q+2)(q+1)(q)}{(k+2)(k+1)(k)(k-1)\cdots(q+6)(q+5)(q+4)(q+3)}\cdot f(q)\\
&= \frac{(q+2)(q+1)q}{(k+2)(k+1)k}\cdot f(q)
\end{aligned}$$

Plugging in the stationary solution for $f(q)$ from Eq. (4.29) gives the general solution

$$f(k) = \frac{(q+2)(q+1)q}{(k+2)(k+1)k}\cdot\frac{2}{(q+2)} = \frac{2q(q+1)}{k(k+1)(k+2)}$$

For constant q and large k, it is easy to see that the degree distribution scales as

$$f(k) \propto k^{-3} \tag{4.31}$$

In other words, the BA model yields a power-law degree distribution with $\gamma = 3$, especially for large degrees.

Continuous Approach The continuous approach is also called the *mean-field* method. In the BA model, the vertices that are added early on tend to have a higher degree, because they have more chances to acquire connections from the vertices that are added to the network at a later time. The time dependence of the degree of a vertex can be approximated as a continuous random variable. Let $k_i = d_t(i)$ denote the degree of vertex v_i at time t. At time t, the probability that the newly added node u links to

v_i is given as $\pi_t(i)$. Further, the change in v_i's degree per time-step is given as $q \cdot \pi_t(i)$. Using the approximation that $n_t \simeq t$ and $m_t \simeq qt$ from Eq. (4.16), the rate of change of k_i with time can be written as

$$\frac{dk_i}{dt} = q \cdot \pi_t(i) = q \cdot \frac{k_i}{2qt} = \frac{k_i}{2t}$$

Rearranging the terms in the preceding equation $\frac{dk_i}{dt} = \frac{k_i}{2t}$ and integrating on both sides, we have

$$\begin{aligned}
\int \frac{1}{k_i} dk_i &= \int \frac{1}{2t} dt \\
\ln k_i &= \frac{1}{2} \ln t + C \\
e^{\ln k_i} &= e^{\ln t^{1/2}} \cdot e^C, \text{ which implies} \\
k_i &= \alpha \cdot t^{1/2}
\end{aligned} \tag{4.32}$$

where C is the constant of integration, and thus $\alpha = e^C$ is also a constant.

Let t_i denote the time when node i was added to the network. Because the initial degree for any node is q, we obtain the boundary condition that $k_i = q$ at time $t = t_i$. Plugging these into Eq. (4.32), we get

$$\begin{aligned}
k_i &= \alpha \cdot t_i^{1/2} = q, \text{ which implies that} \\
\alpha &= \frac{q}{\sqrt{t_i}}
\end{aligned} \tag{4.33}$$

Substituting Eq. (4.33) into Eq. (4.32) leads to the particular solution

$$k_i = \alpha \cdot \sqrt{t} = q \cdot \sqrt{t/t_i} \tag{4.34}$$

Intuitively, this solution confirms the rich-gets-richer phenomenon. It suggests that if a node v_i is added early to the network (i.e., t_i is small), then as time progresses (i.e., t gets larger), the degree of v_i keeps on increasing (as a square root of the time t).

Let us now consider the probability that the degree of v_i at time t is less than some value k, i.e., $P(k_i < k)$. Note that if $k_i < k$, then by Eq. (4.34), we have

$$\begin{aligned}
k_i &< k \\
q \cdot \sqrt{\frac{t}{t_i}} &< k \\
\frac{t}{t_i} &< \frac{k^2}{q^2}, \text{ which implies that} \\
t_i &> \frac{q^2 t}{k^2}
\end{aligned}$$

Thus, we can write

$$P(k_i < k) = P\left(t_i > \frac{q^2 t}{k^2}\right) = 1 - P\left(t_i \le \frac{q^2 t}{k^2}\right)$$

In other words, the probability that node v_i has degree less than k is the same as the probability that the time t_i at which v_i enters the graph is greater than $\frac{q^2}{k^2}t$, which in turn can be expressed as 1 minus the probability that t_i is less than or equal to $\frac{q^2}{k^2}t$.

Note that vertices are added to the graph at a uniform rate of one vertex per time-step, that is, $\frac{1}{n_t} \simeq \frac{1}{t}$. Thus, the probability that t_i is less than or equal to $\frac{q^2}{k^2}t$ is given as

$$\begin{aligned} P(k_i < k) &= 1 - P\left(t_i \le \frac{q^2 t}{k^2}\right) \\ &= 1 - \frac{q^2 t}{k^2} \cdot \frac{1}{t} \\ &= 1 - \frac{q^2}{k^2} \end{aligned}$$

Because v_i is any generic node in the graph, $P(k_i < k)$ can be considered to be the cumulative degree distribution $F_t(k)$ at time t. We can obtain the degree distribution $f_t(k)$ by taking the derivative of $F_t(k)$ with respect to k to obtain

$$\begin{aligned} f_t(k) = \frac{d}{dk} F_t(k) &= \frac{d}{dk} P(k_i < k) \\ &= \frac{d}{dk}\left(1 - \frac{q^2}{k^2}\right) \\ &= 0 - \left(\frac{k^2 \cdot 0 - q^2 \cdot 2k}{k^4}\right) \\ &= \frac{2q^2}{k^3} \\ &\propto k^{-3} \end{aligned} \tag{4.35}$$

In Eq. (4.35) we made use of the quotient rule for computing the derivative of the quotient $f(k) = \frac{g(k)}{h(k)}$, given as

$$\frac{df(k)}{dk} = \frac{h(k) \cdot \frac{dg(k)}{dk} - g(k) \cdot \frac{dh(k)}{dk}}{h(k)^2}$$

Here $g(k) = q^2$ and $h(k) = k^2$, and $\frac{dg(k)}{dk} = 0$ and $\frac{dh(k)}{dk} = 2k$.

Note that the degree distribution from the continuous approach, given in Eq. (4.35), is very close to that obtained from the discrete approach given in Eq. (4.31). Both solutions confirm that the degree distribution is proportional to k^{-3}, which gives the power-law behavior with $\gamma = 3$.

Clustering Coefficient and Diameter

Closed form solutions for the clustering coefficient and diameter for the BA model are difficult to derive. It has been shown that the diameter of BA graphs scales as

$$d(G_t) = O\left(\frac{\log n_t}{\log \log n_t}\right)$$

suggesting that they exhibit *ultra-small-world* behavior, when $q > 1$. Further, the expected clustering coefficient of the BA graphs scales as

$$E[C(G_t)] = O\left(\frac{(\log n_t)^2}{n_t}\right)$$

which is only slightly better than the clustering coefficient for random graphs, which scale as $O(n_t^{-1})$. In Example 4.12, we empirically study the clustering coefficient and diameter for random instances of the BA model with a given set of parameters.

Example 4.12. Figure 4.14 plots the empirical degree distribution obtained as the average of 10 different BA graphs generated with the parameters $n_0 = 3$, $q = 3$, and for $t = 997$ time-steps, so that the final graph has $n = 1000$ vertices. The slope of the line in the log-log scale confirms the existence of a power law, with the slope given as $-\gamma = -2.64$.

The average clustering coefficient over the 10 graphs was $C(G) = 0.019$, which is not very high, indicating that the BA model does not capture the clustering effect. On the other hand, the average diameter was $d(G) = 6$, indicating ultra-small-world behavior.

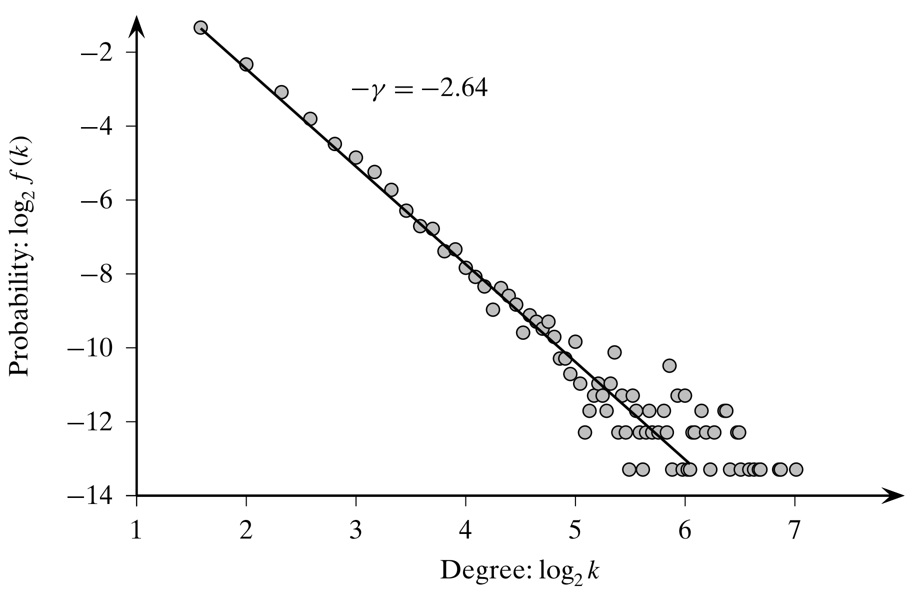


Figure 4.14. Barabási–Albert model ($n_0 = 3, t = 997, q = 3$): degree distribution.

4.5 FURTHER READING

The theory of random graphs was founded in Erdős and Rényi (1959); for a detailed treatment of the topic see Bollobás (2001). Alternative graph models for real-world networks were proposed in Watts and Strogatz (1998) and Barabási and Albert (1999). One of the first comprehensive books on graph data analysis was Wasserman and Faust (1994). More recent books on network science Lewis (2009) and Newman (2010). For PageRank see Brin and Page (1998), and for the hubs and authorities approach see Kleinberg (1999). For an up-to-date treatment of the patterns, laws, and models (including the RMat generator) for real-world networks, see Chakrabarti and Faloutsos (2012).

Barabási, A.-L. and Albert, R. (1999). "Emergence of scaling in random networks." *Science*, 286 (5439): 509–512.

Bollobás, B. (2001). *Random Graphs,* 2nd ed. Vol. 73. New York: Cambridge University Press.

Brin, S. and Page, L. (1998). "The anatomy of a large-scale hypertextual Web search engine." *Computer Networks and ISDN Systems*, 30 (1): 107–117.

Chakrabarti, D. and Faloutsos, C. (2012). "Graph Mining: Laws, Tools, and Case Studies.", *Synthesis Lectures on Data Mining and Knowledge Discovery*, 7(1): 1–207. San Rafael, CA: Morgan & Claypool Publishers.

Erdős, P. and Rényi, A. (1959). "On random graphs." *Publicationes Mathematicae Debrecen*, 6, 290–297.

Kleinberg, J. M. (1999). "Authoritative sources in a hyperlinked environment." *Journal of the ACM*, 46 (5): 604–632.

Lewis, T. G. (2009). *Network Science: Theory and Applications*. Hoboken. NJ: John Wiley & Sons.

Newman, M. (2010). *Networks: An Introduction*. Oxford: Oxford University Press.

Wasserman, S. and Faust, K. (1994). *Social Network Analysis: Methods and Applications*. Structural Analysis in the Social Sciences. New York: Cambridge University Press.

Watts, D. J. and Strogatz, S. H. (1998). "Collective dynamics of 'small-world' networks." *Nature*, 393 (6684): 440–442.

4.6 EXERCISES

Q1. Given the graph in Figure 4.15, find the fixed-point of the prestige vector.

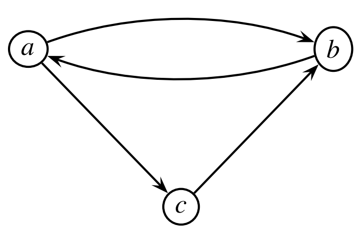


Figure 4.15. Graph for Q1

Q2. Given the graph in Figure 4.16, find the fixed-point of the authority and hub vectors.

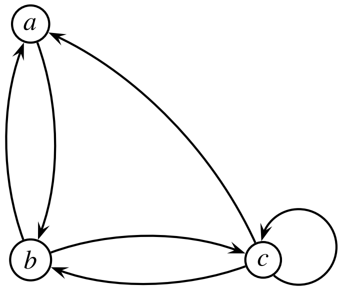


Figure 4.16. Graph for Q2.

Q3. Consider the double star graph given in Figure 4.17 with n nodes, where only nodes 1 and 2 are connected to all other vertices, and there are no other links. Answer the following questions (treating n as a variable).

(a) What is the degree distribution for this graph?
(b) What is the mean degree?
(c) What is the clustering coefficient for vertex 1 and vertex 3?
(d) What is the clustering coefficient $C(G)$ for the entire graph? What happens to the clustering coefficient as $n \to \infty$?
(e) What is the transitivity $T(G)$ for the graph? What happens to $T(G)$ and $n \to \infty$?
(f) What is the average path length for the graph?
(g) What is the betweenness value for node 1?
(h) What is the degree variance for the graph?

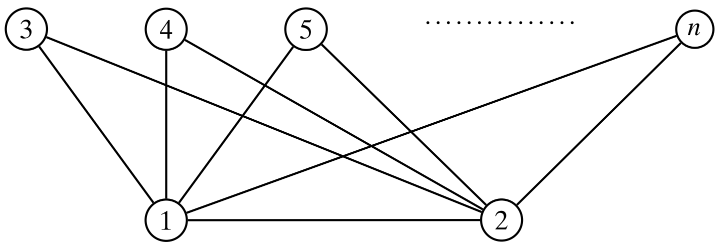


Figure 4.17. Graph for Q3.

Q4. Consider the graph in Figure 4.18. Compute the hub and authority score vectors. Which nodes are the hubs and which are the authorities?

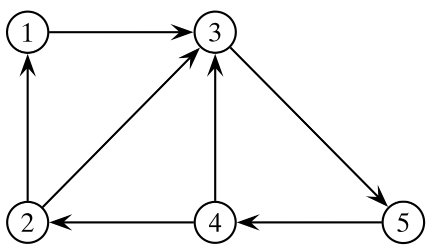


Figure 4.18. Graph for Q4.

Q5. Prove that in the BA model at time-step $t+1$, the probability $\pi_t(k)$ that some node with degree k in G_t is chosen for preferential attachment is given as

$$\pi_t(k) = \frac{k \cdot n_t(k)}{\sum_i i \cdot n_t(i)}$$