we have m integers k_i, $i = 1 \ldots m$, which are independent random numbers drawn from this distribution. For instance, they could be the degrees of m randomly chosen vertices in a network with degree distribution p_k. Then the probability distribution of the sum $\sum_{i=1}^{m} k_i$ of those m integers has generating function $[g(z)]^m$. This is a very powerful result and it is worth taking a moment to see how it arises and what it means.

Given that our integers are independently drawn from the distribution p_k, the probability that they take a particular set of values $\{k_i\}$ is simply $\prod_i p_{k_i}$ and the probability π_s that the values drawn add up to a specific sum s is the sum of these probabilities over all sets $\{k_i\}$ that add up to s:

$$\pi_s = \sum_{k_1=0}^{\infty} \cdots \sum_{k_m=0}^{\infty} \delta\bigl(s, \textstyle\sum_i k_i\bigr) \prod_{i=1}^{m} p_{k_i}, \tag{13.27}$$

where $\delta(a,b)$ is the Kronecker delta. Then the generating function $h(z)$ for the distribution π_s is

$$\begin{aligned} h(z) &= \sum_{s=0}^{\infty} \pi_s z^s \\ &= \sum_{s=0}^{\infty} z^s \sum_{k_1=0}^{\infty} \cdots \sum_{k_m=0}^{\infty} \delta\bigl(s, \textstyle\sum_i k_i\bigr) \prod_{i=1}^{m} p_{k_i} \\ &= \sum_{k_1=0}^{\infty} \cdots \sum_{k_m=0}^{\infty} z^{\sum_i k_i} \prod_{i=1}^{m} p_{k_i} \\ &= \sum_{k_1=0}^{\infty} \cdots \sum_{k_m=0}^{\infty} \prod_{i=1}^{m} p_{k_i} z^{k_i} = \biggl[\sum_{k=0}^{\infty} p_k z^k\biggr]^m \\ &= \bigl[g(z)\bigr]^m. \end{aligned} \tag{13.28}$$

Thus, for example, if we know the degree distribution of a network, it is a straightforward matter to calculate the probability distribution of the sum of the degrees of m randomly chosen vertices from that network. This will turn out to be important in the developments that follow.

13.2 THE CONFIGURATION MODEL

Let us turn now to the main topic of this chapter, the development of the theory of random graphs with general degree distributions.

We can turn the random graph of Chapter 12 into a much more flexible model for networks by modifying it so that the degrees of its vertices are no longer restricted to having a Poisson distribution, and in fact it is possible to

modify the model so as to give the network any degree distribution we please. Just as with the Poisson random graph, which can be defined in several slightly different ways, there is more than one way to define random graphs with general degree distributions. Here we describe two of them, which are roughly the equivalent of the $G(n,m)$ and $G(n,p)$ random graphs of Section 12.1.

The most widely studied of the generalized random graph models is the *configuration model*. The configuration model is actually a model of a random graph with a given degree *sequence*, rather than degree distribution. That is, the exact degree of each individual vertex in the network is fixed, rather than merely the probability distribution from which those degrees are chosen. This in turn fixes the number of edges in the network, since the number of edges is given by Eq. (6.21) to be $m = \frac{1}{2}\sum_i k_i$. Thus this model is in some ways analogous to $G(n,m)$, which also fixes the number of edges. (It is quite simple, however, to modify the model for cases where only the degree distribution is known and not the exact degree sequence. We describe how this is done at the end of this section.)

See Section 8.3 for a discussion of the distinction between degree sequences and degree distributions.

Suppose then that we specify the degree k_i that each vertex $i = 1 \ldots n$ in our network is to take. We can create a random network with these degrees as follows. We give each vertex i a total of k_i "stubs" of edges as depicted in Fig. 13.1. There are $\sum_i k_i = 2m$ stubs in total, where m is the total number of edges. Then we choose two of the stubs uniformly at random and we create an edge by connecting them to one another, as indicated by the dashed line in the figure. Then we choose another pair from the remaining $2m-2$ stubs, connect those, and so on until all the stubs are used up. The end result is a network in which every vertex has exactly the desired degree.

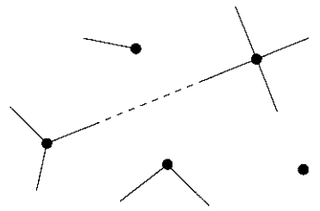

Figure 13.1: The configuration model. Each vertex is given a number of "stubs" of edges equal to its desired degree. Then pairs of stubs are chosen at random and connected together to form edges (dotted line).

More specifically the end result is a particular *matching* of the stubs, a particular set of pairings of stubs with other stubs. The process above generates each possible matching of stubs with equal probability. Technically the configuration model is defined as the ensemble in which each matching with the chosen degree sequence appears with the same probability (those with any other degree sequence having probability zero), and the process above is a process for drawing networks from the configuration model ensemble.

The uniform distribution over matchings in the configuration model has the important consequence that any stub in a configuration model network is equally likely to be connected to any other. This, as we will see, is the crucial property that makes the model solvable for many of its properties.

There are a couple of minor catches with the network generation process described here. First, there must be an even number of stubs overall if we want to end up with a network consisting only of vertices and edges, with no dangling stubs left over. This means that the sum $\sum_i k_i$ of the degrees must add up to an even number. We will assume that the degrees we have chosen satisfy this condition, otherwise it is clearly not possible to create a graph with the given degree sequence.

A second issue is that the network may contain self-edges or multiedges, or both. There is nothing in the network generation process that prevents us from creating an edge that connects a vertex to itself or that connects two vertices that are already connected by another edge. One might imagine that one could avoid this by rejecting the creation of any such edges during the process, but it turns out that this is not a good idea. A network so generated is no longer drawn uniformly from the set of possible matchings, which means that properties of the model can no longer be calculated analytically, at least by any means currently known. It can also mean that the network creation process breaks down completely. Suppose, for example, that we come to the end of the process, when there are just two stubs left to be joined, and find that those two both belong to the same vertex so that joining them would create a self-edge. Then either we create the self-edge or the network generation process fails.

In practice, therefore, it makes more sense to allow the creation of both multiedges and self-edges in our networks and the standard configuration model does so. Although some real-world networks have self-edges or multiedges in them, most do not, and to some extent this makes the configuration model less satisfactory as a network model. However, as shown below, the average number of self-edges and multiedges in the configuration model is a constant as the network becomes large, which means that the density of self-edges and multiedges tends to zero in this limit. This means, to all intents and purposes, that we can ignore the self-edges and multiedges in the large size limit.[3]

A further issue with the configuration model is that, while all matchings of stubs appear with equal probability in the model, that does not mean that all *networks* appear with equal probability because more than one matching can correspond to the same network, i.e., the same topological connections between vertices. If we label the stubs to keep track of which is which, then

[3]Even for finite-sized networks the difference between the properties of a configuration model network and a similar network without self-edges and multiedges would only result in a correction of order $1/n$ into our results. For the large networks that are the focus of most modern network studies this means that the error introduced by allowing self-edges and multiedges is small.

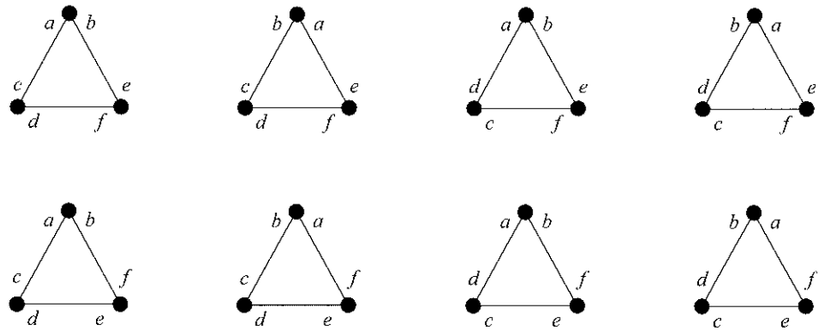

Figure 13.2: Eight stub matchings that all give the same network. This small network is composed of three vertices of degree two and hence having two stubs each. The stubs are lettered to identify them and there are two distinct permutations of the stubs at each vertex for a total of eight permutations overall. Each permutation gives rise to a different matching of stub to stub but all matchings correspond to the same topological configuration of edges, and hence there are eight ways in which this particular configuration can be generated by the stub matching process.

there are typically many different ways we can join up pairs of labeled stubs to create the same final configuration of edges. Figure 13.2 shows an example of a set of eight matchings that all correspond to the same three-vertex network.

In general, one can generate all the matchings that correspond to a given network by taking any one matching for that network and permuting the stubs at each vertex in every possible way. Since the number of permutations of the k_i stubs at a vertex i is $k_i!$, this implies that the number of matchings corresponding to each network is $N(\{k_i\}) = \prod_i k_i!$, which takes the same value for all networks, since the degrees are fixed. This implies that in fact networks occur with equal probability in the configuration model: if there are $\Omega(\{k_i\})$ matchings, each occurring with the same probability, then each *network* occurs with probability N/Ω.

However, this is not completely correct. If a network contains self-edges or multiedges then not all permutations of the stubs in the network result in a new matching of stubs. Consider Fig. 13.3. Panel (a) shows a network with the same degree sequence as those of Fig. 13.2, but a different matching of the stubs that creates a network with one self-edge and a multiedge consisting of two parallel single edges. In panel (b) we have permuted the stubs a and b at the ends of the self-edge but, as we can see, this has not resulted in a new matching of the stubs themselves. Stubs a and b are still connected to one another just as they were before. (The network is *drawn* differently now, but in terms of the matching and the topology of the edges nothing has changed

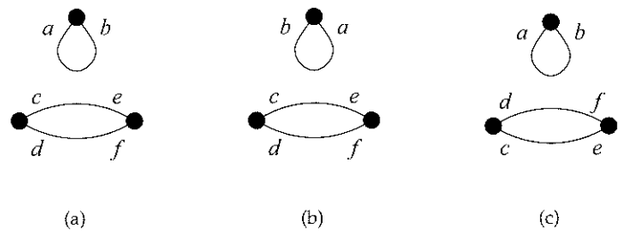

Figure 13.3: Permutations that do not produce new matchings. (a) The network shown here has the same degree sequence as those of Fig. 13.2 but a different configuration of edges, having one self-loop and a multiedge consisting of two parallel edges. (b) If we permute the stubs a and b of the self-edge we do not generate a new matching, because a is still matched with b, just as before. (c) If we permute the stubs at either end of a multiedge in exactly the same way we do not generate a new matching, since each stub at one end of the multiedge is still matched with the same stub at the other end.

from panel (a).) In panel (c) we have identically permuted the stubs at both ends of the multiedge. Again this has no effect on which stubs are matched with which others.

In general, for each multiedge in a network a permutation of the stubs at one end fails to generate a new matching if we simultaneously permute the stubs at the other end in the same way. This means that the total number of matchings is reduced by a factor of $A_{ij}!$, since A_{ij} is equal to the multiplicity of the edge between i and j. Indeed, this expression is correct even for vertex pairs not connected by a multiedge, if we adopt the convention that $0! = 1$. For self-edges there is a further factor of two because the interchange of the two ends of the edge does not generate a new matching. Combining these results, the number of matchings corresponding to a network turns out to be

$$N = \frac{\prod_i k_i!}{\prod_{i<j} A_{ij}! \prod_i A_{ii}!!}, \tag{13.29}$$

where $n!! = n(n-2)(n-4)\ldots 2$ with n even is the so-called double factorial of n. Then the total probability of a particular network within the configuration model ensemble is N/Ω as before. Since the denominator in Eq. (13.29) depends not only on the degree sequence but also on the structure of the network itself, different networks do appear with different probabilities.

As we mentioned, however, the average densities of self-edges and multiedges in the configuration model vanish as n becomes large, so that the variation in probabilities is relatively small in the large-n limit, but it nonetheless does occasionally assume some importance and is therefore worth bearing in mind (see, for instance, Ref. [220]).

As discussed above, we are sometimes (indeed often) interested in the case where it is the degree distribution of the network that is specified rather than the degree sequence. That is, we specify the probability distribution p_k from which the degree sequence is drawn rather than the sequence itself. We can define an obvious extension of the configuration model to this case: we draw a degree sequence from the specified distribution and then generate a network with that degree sequence using the technique described above. More precisely, we define an ensemble in which each degree sequence $\{k_i\}$ appears with probability $\prod_i p_{k_i}$. Then if we can calculate an average value $X(\{k_i\})$ for some quantity of interest X in the standard configuration model, the average value in the extended model is given by

$$\langle X \rangle = \sum_{k_1=0}^{\infty} \cdots \sum_{k_n=0}^{\infty} X(\{k_i\}) \prod_{i=0}^{n} p_{k_i}. \tag{13.30}$$

In practice the difference between the two models is not actually very great. As we will see, the crucial parameter that enters into most of our configuration model calculations is the fraction of vertices that have each possible degree k. In the extended model above, this fraction is, by definition, equal to p_k in the limit of large n. If, on the other hand, the degree sequence is fixed then we simply calculate the fraction from the degree sequence and then use those numbers. In either the case the formulas for calculated quantities are the same.

13.2.1 EDGE PROBABILITY IN THE CONFIGURATION MODEL

A central property of the configuration model is the probability p_{ij} of the occurrence of an edge between two specified vertices, i and j. Obviously if either vertex i or vertex j has degree zero then the probability of an edge is zero, so let us assume that $k_i, k_j > 0$. Now consider any one of the stubs that emerges from vertex i. What is the probability that this stub is connected by an edge to any of the stubs of vertex j? There are $2m$ stubs in total, or $2m-1$ excluding the one connected to i that we are currently looking at. Of those $2m-1$, exactly k_j of them are attached to vertex j. So, given that any stub in the network is equally likely to be connected to any other, the probability that our particular stub is connected to any of those around vertex j is $k_j/(2m-1)$. But there are k_i stubs around vertex i, so the total probability of a connection between i and j is

$$p_{ij} = \frac{k_i k_j}{2m-1}. \tag{13.31}$$

Technically, since we have added the probabilities of independent events, this is really the average number of edges between i and j, rather than the probability of having an edge at all. But in the limit of large m, this number becomes small (for given k_i, k_j), and the average number of edges and the probability of an edge become equal. Also in the limit of large m we can ignore the -1 in the denominator and hence we can write

$$p_{ij} = \frac{k_i k_j}{2m}. \tag{13.32}$$

Note that, even though we assumed $k_i, k_j > 0$, this expression also gives the right result if either degree is zero, namely that in that case the probability of connection is zero.

We can use this result, for example, to calculate the probability of having two edges between the same pair of vertices. The probability of having one edge between vertices i and j is p_{ij} as above. Once we have one edge between the vertices the number of available stubs at each is reduced by one, and hence the probability of having a second edge is given by Eq. (13.32) but with k_i and k_j each reduced by one: $(k_i - 1)(k_j - 1)/2m$. Thus the probability of having (at least) two edges, i.e., of having a multiedge between i and j, is $k_i k_j (k_i - 1)(k_j - 1)/(2m)^2$ and, summing this probability over all vertices and dividing by two (to avoid double counting of vertex pairs), we find that the expected total number of multiedges in the network is

$$\begin{aligned}\frac{1}{2(2m)^2}\sum_{ij} k_i k_j (k_i - 1)(k_j - 1) &= \frac{1}{2\langle k\rangle^2 n^2}\sum_i k_i(k_i - 1)\sum_j k_j(k_j - 1) \\ &= \frac{1}{2}\left[\frac{\langle k^2\rangle - \langle k\rangle}{\langle k\rangle}\right]^2,\end{aligned} \tag{13.33}$$

where

$$\langle k\rangle = \frac{1}{n}\sum_i k_i, \qquad \langle k^2\rangle = \frac{1}{n}\sum_i k_i^2, \tag{13.34}$$

and we have used $2m = \langle k\rangle n$ (see Eq. (6.23)). Thus the expected number of multiedges remains constant as the network grows larger, so long as $\langle k^2\rangle$ is constant and finite, and the density of multiedges—the number per vertex—vanishes as $1/n$. We used this result in a number of our earlier arguments.[4]

Another way to derive the expression in Eq. (13.32) is to observe that there are $k_i k_j$ possible edges we could form between vertices i and j, while the total number of possible edges in the whole graph is the number of ways of

[4]For networks with power-law degree distributions $\langle k^2\rangle$ diverges, as described in Section 8.4.2, and in that case the density of multiedges may not vanish or may do so more slowly than $1/n$.

choosing a pair of stubs from the $2m$ total stubs, or $\binom{2m}{2} = m(2m-1)$. The probability that any particular edge falls between i and j is thus given by the ratio $k_i k_j / m(2m-1)$, and if we make a total of m edges then the expected total number of edges between i and j is m times this quantity, which gives us Eq. (13.31) again.

The only case in which this derivation is not quite right is for self-edges. In that case the number of pairs of stubs is not $k_i k_j$ but instead is $\binom{k_i}{2} = \frac{1}{2}k_i(k_i - 1)$ and hence the probability of a self-edge from vertex i to itself is

$$p_{ii} = \frac{k_i(k_i - 1)}{4m}. \tag{13.35}$$

We can use this result to calculate the expected number of self-edges in the network, which is given by the sum over all vertices i:

$$\sum_i p_{ii} = \sum_i \frac{k_i(k_i - 1)}{4m} = \frac{\langle k^2\rangle - \langle k\rangle}{2\langle k\rangle}, \tag{13.36}$$

This expression remains constant as $n \to \infty$ provided $\langle k^2\rangle$ remains constant, and hence, as with the multiedges, the density of self-edges in the network vanishes as $1/n$ in the limit of large network size.

We can use Eqs. (13.32) and (13.35) to calculate a number of other properties of vertices in the configuration model. For instance, we can calculate the expected number n_{ij} of common neighbors that vertices i and j share. The probability that i is connected to another vertex l is p_{il} and the probability that j is connected to the same vertex would likewise normally be p_{jl}. However, as with the calculation of multiedges above, if we already know that i is connected to l, then the number of available stubs at vertex l is reduced by one and, rather than being given by the normal expression (13.32), the probability of a connection between j and l is $k_j(k_l - 1)/2m$. Multiplying the probabilities for the two edges and summing over l, we then get our expression for the expected number of common neighbors of i and j:

$$\begin{aligned} n_{ij} &= \sum_l \frac{k_i k_l}{2m}\,\frac{k_j(k_l - 1)}{2m} = \frac{k_i k_j}{2m}\,\frac{\sum_l k_l(k_l - 1)}{n\langle k\rangle} \\ &= p_{ij}\,\frac{\langle k^2\rangle - \langle k\rangle}{\langle k\rangle}. \end{aligned} \tag{13.37}$$

Thus the probability of sharing a common neighbor is equal to the probability $p_{ij} = k_i k_j / 2m$ of having a direct connection times a multiplicative factor that depends only on the mean and variance of the degree distribution but not on the properties of the vertices i and j themselves.

In this calculation we have ignored the fact that the probability of self-edges, Eq. (13.35), is different from the probability for other edges. As we have seen, however, the density of self-edges in the configuration model tends to zero as $n \to \infty$, so in that limit it is usually safe to make the approximation that Eq. (13.32) applies for all i and j.

13.2.2 RANDOM GRAPHS WITH GIVEN EXPECTED DEGREE

The configuration model of the previous section is, as we have said, similar in some ways to the standard random graph $G(n,m)$ described in Section 12.1, in which we distribute a fixed number m of edges at random between n vertices. In the configuration model the total number of edges is again fixed, having value $m = \frac{1}{2}\sum_i k_i$, but in addition we now also fix the individual degree of every vertex as well.

It is natural to ask whether there is also an equivalent of $G(n,p)$—the model in which only the probability of edges is fixed and not their number—and indeed there is. We simply place an edge between each pair of vertices i,j with independent probabilities taking the form of Eq. (13.32). We define a parameter c_i for each vertex and then place an edge between vertices i and j with probability $p_{ij} = c_i c_j/2m$. As with the configuration model, we must allow self-edges if the model is to be tractable, and again self-edges have to be treated a little differently from ordinary edges. It turns out that the most satisfactory definition of the edge probability is[5]

$$p_{ij} = \begin{cases} c_i c_j/2m & \text{for } i \neq j, \\ c_i^2/4m & \text{for } i = j, \end{cases} \tag{13.38}$$

where m is now defined by[6]

$$\sum_i c_i = 2m. \tag{13.39}$$

[5]As before, p_{ij} should really be regarded as the expected number of edges between i and j rather than the probability and in fact the proper formulation of the model is that we place a Poisson-distributed number of edges with mean p_{ij} between each pair of vertices i,j. Thus the model can in principle have multiedges as well as self-edges, just as in the configuration model. In the limit of large m and constant c_i, however, the probability and the expected number again become equal, and the density of multiedges tends to zero, so the distinction is unimportant.

[6]Another way of putting this is that the average value $\langle A_{ij}\rangle$ of an element of the adjacency matrix is simply $\langle A_{ij}\rangle = c_i c_j/2m$ for all i,j—recall that the diagonal element A_{ii} of the adjacency matrix is defined to be twice the number of self-edges at vertex i, and this compensates for the extra factor of two in Eq. (13.38).

With this choice the average number of edges in the network is

$$\sum_{i\le j} p_{ij} = \sum_{i<j} \frac{c_i c_j}{2m} + \sum_i \frac{c_i^2}{4m} = \sum_{ij} \frac{c_i c_j}{4m} = m, \tag{13.40}$$

as before. We can also calculate the average number of ends of edges connected to a vertex i, i.e., its average degree $\langle k_i\rangle$. Allowing for the fact that a self-edge contributes two ends of edges to the degree, we get

$$\langle k_i\rangle = 2p_{ii} + \sum_{j(\neq i)} p_{ij} = \frac{c_i^2}{2m} + \sum_{j(\neq i)} \frac{c_i c_j}{2m} = \sum_j \frac{c_i c_j}{2m} = c_i. \tag{13.41}$$

In other words the parameters c_i appearing in the definition of p_{ij}, Eq. (13.38), are the average or expected degrees in this model, just as the parameter c in $G(n,p)$ is the average degree of a vertex. The *actual* degree of a vertex could in principle take almost any value, depending on the luck of the draw about which edges happen to get randomly created and which do not. In fact one can show that the degree of vertex i will have a Poisson distribution with mean c_i, meaning that in practice it will be quite narrowly distributed about c_i, but there will certainly be some variation, unless c_i is zero.[7] Note that c_i does not have

[7]The probabilities of edges between vertex i and each other vertex are independent, which immediately implies that the degree has a Poisson distribution. This may be obvious to you—if you're a statistician, for example—but if not, here is a proof, which makes use of generating functions.

The probability that there are edges connecting vertex i to any specific set of vertices, including itself, is given by a product of factors p_{ij} for each edge present and $(1-p_{ij})$ for each edge not present. This product can conveniently be written in the form

$$p_{ii}^{A_{ii}/2}(1-p_{ii})^{1-A_{ii}/2} \prod_{j(\neq i)} p_{ij}^{A_{ij}}(1-p_{ij})^{1-A_{ij}},$$

where A_{ij} is the standard adjacency matrix and we adopt the convention that $0^0 = 1$ for any cases where $p_{ij} = 0$. Note that it is important to separate out the term for p_{ii} as shown, since it takes a slightly different form from the others. Recall that a self-edge is represented by a diagonal element $A_{ii} = 2$ in the adjacency matrix (see Section 6.2) and we must allow for this with the factors of two above.

The probability $p_k^{(i)}$ that vertex i has degree exactly k is the sum of these probabilities over all cases where there the ith row of the adjacency matrix adds up to k (including the 2s that appear for self-edges, since a self-edge contributes +2 to the degree). We can write this sum as

$$p_k^{(i)} = \sum_{A_{i1}=0,1} \cdots \sum_{A_{ii}=0,2} \cdots \sum_{A_{in}=0,1} \delta\Big(k, \textstyle\sum_j A_{ij}\Big) p_{ii}^{A_{ii}/2}(1-p_{ii})^{1-A_{ii}/2} \prod_{j(\neq i)} p_{ij}^{A_{ij}}(1-p_{ij})^{1-A_{ij}},$$

where $\delta(a,b)$ is the Kronecker delta. It is tricky to evaluate this sum directly because of the constraint imposed by the delta function, but we can do it using a generating function. Multiplying both sides of the equation by z^k, summing over all k, and defining the generating function $g_i(z) = \sum_k p_k^{(i)} z^k$, we get

to be an integer, unlike the degrees k_i appearing in the configuration model.

Thus in this model we specify the expected number of edges m and the expected degree sequence $\{c_i\}$ of the network but not the actual number of edges and actual degree sequence. This is again analogous to $G(n,p)$, in which we specify only the expected number of edges and not the actual number. Unfortunately, this means we usually cannot choose the degree *distribution* of our network, because the distribution of the actual degrees k_i is not the same as the distribution of the expected degrees c_i. This is a substantial disadvantage

---

$$\begin{aligned}
g_i(z) &= \sum_{k=0}^{\infty} z^k \sum_{A_{i1}=0,1} \cdots \sum_{A_{ii}=0,2} \cdots \sum_{A_{in}=0,1} \delta\Big(k, \textstyle\sum_j A_{ij}\Big) p_{ii}^{A_{ii}/2} (1-p_{ii})^{1-A_{ii}/2} \prod_{j(\neq i)} p_{ij}^{A_{ij}} (1-p_{ij})^{1-A_{ij}} \\
&= \sum_{A_{i1}=0,1} \cdots \sum_{A_{ii}=0,2} \cdots \sum_{A_{in}=0,1} z^{\sum_j A_{ij}} p_{ii}^{A_{ii}/2} (1-p_{ii})^{1-A_{ii}/2} \prod_{j(\neq i)} p_{ij}^{A_{ij}} (1-p_{ij})^{1-A_{ij}} \\
&= \sum_{A_{i1}=0,1} \cdots \sum_{A_{ii}=0,2} \cdots \sum_{A_{in}=0,1} (p_{ii}z^2)^{A_{ii}/2} (1-p_{ii})^{1-A_{ii}/2} \prod_{j(\neq i)} (p_{ij}z)^{A_{ij}} (1-p_{ij})^{1-A_{ij}} \\
&= (1-p_{ii}+p_{ii}z^2) \prod_{j(\neq i)} (1-p_{ij}+p_{ij}z) \\
&= \Big[1+\frac{c_i^2}{4m}(z^2-1)\Big] \prod_{j(\neq i)} \Big[1+\frac{c_i c_j}{2m}(z-1)\Big].
\end{aligned}$$

Taking logs of both sides and going to the limit of large size, where $m \to \infty$ (with the c_i remaining finite), we then get

$$\begin{aligned}
\ln g_i(z) &= \lim_{m\to\infty} \Big\{ \ln\Big[1+\frac{c_i^2}{4m}(z^2-1)\Big] + \sum_{j(\neq i)} \ln\Big[1+\frac{c_i c_j}{2m}(z-1)\Big] \Big\} \\
&= \frac{c_i^2}{4m}(z^2-1) + \sum_{j(\neq i)} \frac{c_i c_j}{2m}(z-1) \\
&= \frac{c_i^2}{4m}(z^2-1) - \frac{c_i^2}{2m}(z-1) + \sum_{j=1}^{n} \frac{c_i c_j}{2m}(z-1) \\
&= \frac{c_i^2}{4m}(z^2-1) - \frac{c_i^2}{2m}(z-1) + c_i(z-1) \\
&= c_i(z-1)\Big[1+\frac{c_i(z-1)}{4m}\Big],
\end{aligned}$$

where we have made use of Eq. (13.39) in the second-to-last line. For large m, the second term in the square brackets becomes negligible compared to the first and, taking exponentials again,

$$g_i(z) = \mathrm{e}^{c_i(z-1)}.$$

Now we can derive the probability distribution of the degree of vertex i by differentiating:

$$p_k^{(i)} = \frac{1}{k!} \frac{\mathrm{d}^k g_i}{\mathrm{d}z^k}\bigg|_{z=0} = \mathrm{e}^{-c_i} \frac{c_i^k}{k!},$$

which is indeed a Poisson distribution, with mean c_i, as promised.

of the model since the degree distribution is widely considered to be a crucial property of networks.[8]

This is unfortunate, because this model is in other respects a very nice one. It is straightforward to treat analytically and many of the derivations are substantially simpler for this model than for the configuration model. Nonetheless, because we place such a premium on being able to choose the degree distribution, this model is in fact hardly ever used in real calculations of the properties of networks. Instead, most calculations are made using the configuration model and this is the direction that we will take in this book as well. In the following sections, we describe how one can make use of the machinery of generating functions to calculate many of the properties of the configuration model exactly in the limit of large network size.

13.3 EXCESS DEGREE DISTRIBUTION

In the remainder of this chapter we describe the calculation of a variety of properties of the configuration model. We begin our discussion with some fundamental observations about the model—and networks in general—that will prove central to later developments.

Consider a configuration model with degree distribution p_k, meaning that a fraction p_k of the vertices have degree k. (We can consider either the standard version of the model in which the degree sequence is fixed, as in Section 13.2, or the version of Eq. (13.30) in which only the distribution is fixed but not the exact degree sequence.) The distribution p_k tells us the probability that a vertex chosen uniformly at random from our network has degree k. But suppose instead that we take a vertex (randomly chosen or not) and follow one of its edges (assuming it has at least one) to the vertex at the other end. What is the probability that this vertex will have degree k?

The answer cannot just be p_k. For instance, there is no way to reach a vertex with degree zero by following an edge in this way, because a vertex with degree zero has no edges. So the probability of finding a vertex of degree zero is itself zero, and not p_0.

In fact, the correct probability for general k is not hard to calculate. We know that an edge emerging from a vertex in a configuration model network has equal chance of terminating at any "stub" of an edge anywhere else in the network (see Section 13.2). Since there are $\sum_i k_i = 2m$ stubs in total, or $2m-1$

---

[8] It is easy to see that there are some degree distributions that the model cannot reproduce at all—any distribution for which p_k is exactly zero for any k, for instance, since there is always a non-zero probability that any vertex can have any degree.