# Structural and dynamical properties of complex networks

by

Gourab Ghoshal

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Physics)
in The University of Michigan
2009

I guess I should warn you, if I turn out to be particularly clear, you've probably misunderstood what I said.
—Alan Greenspan, 1988 speech, as quoted in The New York Times, October 2005

To see a world in a grain of sand,
And a heaven in a wild flower,
Hold infinity in the palm of your hand,
And eternity in an hour…
—William Blake, *Auguries of Innocence*

To my parents

## ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Mark Newman for his diligent guidance over the past five years. Mark's unique insight into various research problems, coupled with his sense of excellence and sharp wit, has been a source of great inspiration. As a direct academic descendant, I hope to employ the lessons learnt under his tutelage in a constructive fashion. I'd also like to thank Dr. Leonard Sander, Dr. Michal Zochowski, Dr. Lada Adamic and Dr. Cagliyan Kurak for serving on my dissertation committee.

Special thanks to Brian Karrer, Petter Holme, Lada Adamic, Bob Ziff and an anonymous referee, whose often harsh but richly deserved comments on earlier versions of this manuscript have greatly improved the final product. Extra special thanks to my once and perhaps future officemates, Juyong Park, Petter Holme, Elizabeth Leicht, Brian Karrer and Beth Percha who have proven themselves to be excellent colleagues, friends and fellow-procrastinators. A collective thanks goes out to the faculty, staff and students of the Department of Physics, for providing me with a stimulating and exciting environment to work in, as well as contributing to a very enjoyable stay at Ann Arbor. Finally, I would like to thank my parents, to whom I owe all.

The work presented in this dissertation was funded through a combination of grants DMS-0405348, PHY-0200909 and DMS-0804778 from the National Science Foundation, as well as support from the James McDonnell foundation.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

**CHAPTER I**

# Introduction

A network is a mathematical object consisting of a set of points (called vertices or nodes) that are connected to each other in some fashion by lines (called edges). There are a number of systems in the real world that match this description, ranging from technological ones such as the Internet and World Wide Web, biological networks such as that of connections of the nervous systems or blood vessels, food webs, protein interactions, infrastructural systems such as networks of roads and the power-grid, to patterns of social acquaintance such as friendship, network of hollywood actors, connections between business houses and many many more. Research involving networks in some form of the other can be traced back as early as the beginning of the 20th century[1]—where it was mostly the domain of mathematicians and social scientists—leading upto current developments where it is now the subject of scrutiny of a bewildering array of researchers from fields as diverse as biology, ecology, economics, computer science and physics, to name just a few.

The involvement of physicists seems to have generated some mixed reactions. Duncan Watts, a noted social scientist has made the following pithy observation in his popular account on network research *Six Degrees* [148],

[1]Well, actually as early as the 18th century as we will see shortly, however, network research here is meant in terms of analyzing real world networks, whereas previous developments (largely) dealt with theoretical mathematical objects

> Physicists, it turns out, are almost perfectly suited to invading other people's disciplines, being not only extremely clever but also generally much less fussy than most about the problems they choose to study. Physicists tend to see themselves as the lords of the academic jungle, loftily regarding their own methods as above the ken of anybody else and jealously guarding their own terrain. But their alter egos are closer to scavengers, happy to borrow ideas and techniques from anywhere if they seem like they might be useful, and delighted to stomp all over someone else's problem. As irritating as this attitude can be to everybody else, the arrival of physicists into a previously non-physics area of research often presages a period of great discovery and excitement. Mathematicians do the same thing occasionally, but no one descends with such fury and in such great a number as a pack of hungry physicists, adrenalized by the scent of a new problem.

Whether the above is a frank assessment or a backhanded compliment is a matter of debate, nevertheless it is indeed a fact that the involvement of physicists (especially over the last decade) in network research has led to a lot of progress in the field both in theoretical as well as empirical terms. As a fellow pack member of this group, the burden is on the author to continue this trend, and it is hoped the material presented in this dissertation will have contributed to the phenomena of "great discovery and excitement ..." by presenting research on various aspects on the structural and dynamical properties of networks, often (but not always) drawing on methods and concepts inspired by developments in physics.

We will begin by providing a brief overview of some of the important historical contributions in network research leading upto the 1990s, following which we will then motivate why recent developments have led physicists to "descend[ing] with such fury and in such great a number…”. In Sections 1.3 and 1.4 , we will describe the various types of network representations that researchers use to mimic systems in the real world, as well as the standard metrics and mathematical tools employed to investigate their properties. Sections 1.6–1.7 will provide a brief description of the theoretical developments on which most of the work in this dissertation is based. Finally in Sec. 1.8 we will outline the content of the remaining chapters of this dissertation.

## 1.1 Historical overview

One of the first instances of research related to networks, can be found in Leonhard Euler's 1735 work *Solutio problematis ad geometriam situs pertinentis* (The solution of a problem relating to the geometry of position), in which he presented a resolution to the “Seven bridges of Königsberg problem”[2] . The city of Königsberg in Prussia (present day Kaliningrad, Russia) was set on both sides of the Pregel River, and included two large islands which were connected to each other and the mainland by seven bridges. The problem was to find a walk through the city that would cross each bridge once and only once. The islands could not be reached by any route other than the bridges, and every bridge must have been crossed completely every time. Euler's genius was to reformulate the problem in abstract terms, eliminating all features except the list of landmasses and the bridges connecting them. In modern jargon, one replaces each landmass with an abstract vertex (or node), and each bridge with an abstract connection, an edge, which only serves to record which pair of vertices (landmasses) is connected by that bridge. The resulting mathematical structure is called a graph. Euler's formulation of the problem in these terms and his subsequent

[2]While the work was presented to the St. Petersburg Academy in 1735, it was actually published in 1741 in the journal *Commentarii academiae scientiarum Petropolitanae.*

solution (to put it briefly, no, one cannot cross the bridges as proposed) laid the foundations of what is now known as mathematical graph theory, which has been, and to a certain extent still is, the primary analytical method used to study the property of networks.

In the 19th century, further advances in the field of graph theory were carried out by the likes of Thomas Kirkman, William Hamilton (of Hamiltonian fame), Gustav Kirchoff (who employed graph theoretical ideas in the calculation of electrical currents in circuits), Arthur Cayley and Alexander Polya. Although the developments that emerged from their work—primarily topological measures of various graph properties—are widely employed in contemporary research in networks, strictly speaking, their inspiration and focus were more driven by theoretical mathematical objects, rather than systems that conform to the real world (the study of which is what we really mean by network theory in the modern context).

Arguably, the first group of academics to study real-world networks, were social scientists. As early as the 1930s sociologists had speculated that the patterns of connections between individuals in society must somehow relate to their functioning in general. Consequently they embarked on a series of quantitative studies to discern this connection. Notable among the earlier efforts is the work of the Austro-American psychiatrist and sociologist Jacob Moreno who pioneered the systematic recording and analysis of social interaction in small groups, especially classrooms and work groups, believing that the pattern of social connections had some observable structure, and by drawing them on a piece of paper and looking at them, one could discern that structure and thus make some prediction of the functioning of the group [105]. The networks he constructed (which he called *sociograms*) consisted of vertices that represented individuals, with edges connecting any two of them if they

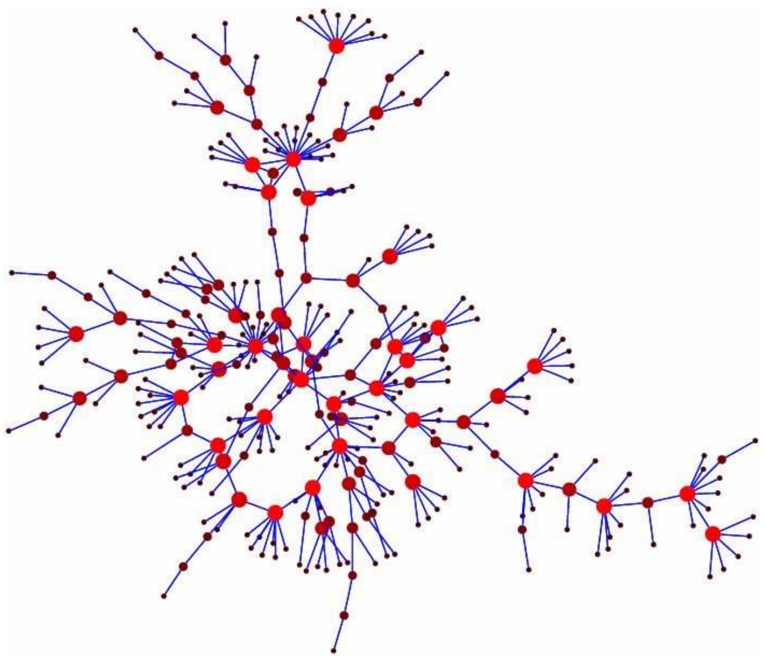

Figure 1.1: An epidemic network of HIV transmission, generated from the data of Potterat *et al.* [125]. Vertices are individuals, and an edge between any two represents transmission of HIV (either through sexual contact or intravenous drug usage). *Picture courtesy of Mark Newman.*

professed to have some kind of acquaintance. Studies along similar lines, were conducted about the same time by Davis *et al.* [44] who examined social circles among women in the American south, while a Harvard group led by W. Lloyd Warner and Elton Mayo explored interpersonal relations at work, such as in the case of Chicago factory workers [130].

The general method that social scientists used to gather data was by directly querying participants, through the use of surveys and questionnaires. For example, if one wanted to construct a network of friendships, first a list of participants had to be made and then each individual in the list had to be asked who they considered a friend. As one can imagine, this was a fairly labor-intensive effort and thus the networks that were studied were fairly limited in size, ranging from tens to at most hundreds of vertices. In such a setting, researchers focussed on the properties of individual vertices and their pattern of connections. Typically studies addressed the issue of centrality (in the sense of which individuals have the most connections or influence in the group) or connectivity (how individual are connected or not to others). A modern version of a social network with size comparable to those studied at the time is shown in Fig. 1.1.

The quantitative studies undertaken by social scientists in turn led to further advances in mathematical techniques. The mathematical psychologist Anatol Rapoport suggested that the frequency distribution of links incident on vertices is an important measure in determining the properties of networks. Along with R. Solomonoff, he proposed a simple model for a network, known as the random graph [139], which was independently studied by the prolific mathematician Paul Erdős and his collaborator Alfréd Rényi [56]. In the model a network is created by placing undirected edges between $n$ fixed vertices in which each of the possible $\binom{n}{2}$ edges are present independently with some probability $p$. We will describe the details of this model in detail in Sec. 1.5.1.

At about the same time, the physicist and historian Derek de Solla Price, was studying citation networks. In a citation network, vertices represent academic publications, and if a particular paper cites another paper, then there is an edge between them. Price in part was continuing already existing work on publication patterns, starting from the pioneering work of Alfred Lotka who in 1926 formulated his “Law of Scientific Productivity”, which states that the distribution of scientific publications follows a power-law [100]. In other words the number of scientists who have published $k$ papers falls of as $k^{-\alpha}$ for some constant exponent $\alpha$. Price was the first to study this in network form and he found that both the number of papers that receive citations, as well as the number of papers a particular paper cites also follow power-laws. This discovery had far reaching implications and inspired future developments in the field, which we will discuss in detail in Sec. 1.6.1.

Based on these developments a number of other researchers from different fields took up the mantle and started to study systems as networks, however the primary impetus until the 1990s was still provided by social scientists who continued to make impressive contributions to the field—see the book by Wasserman and Faust for a review [147]. A notable exception is the early work on biological networks done by Stuart Kauffman who studied genetic regulatory networks, so-called *boolean nets* [87, 88], which greatly influenced future studies in the field.

## 1.2 Interest to Physicists

The research on real-world networks described upto this point shared some common salient features. Most networks studied were limited in size, with the most extreme cases consisting of a few hundreds of vertices. Moreover, the standard procedure was to study a single instance of a network and focus on the individual properties of vertices or edges.

With the 1990s, however, came the availability of widespread computing resources—in particular reasonably cheap and powerful desktop computers—as well as the emergence of large scale communication systems such as the Internet and World Wide Web, that were eminently well suited to be modeled as networks. The Web in particular, being simultaneously a technological, information and social network garnered much interest from a diverse background of researchers. However, the change in scale (from hundreds of vertices to millions or potentially billions), presented a set of new challenges.

Consider the image shown in Fig. 1.2. This is a visualization of the Internet at the autonomous systems level—groups of computers each representing hundreds of

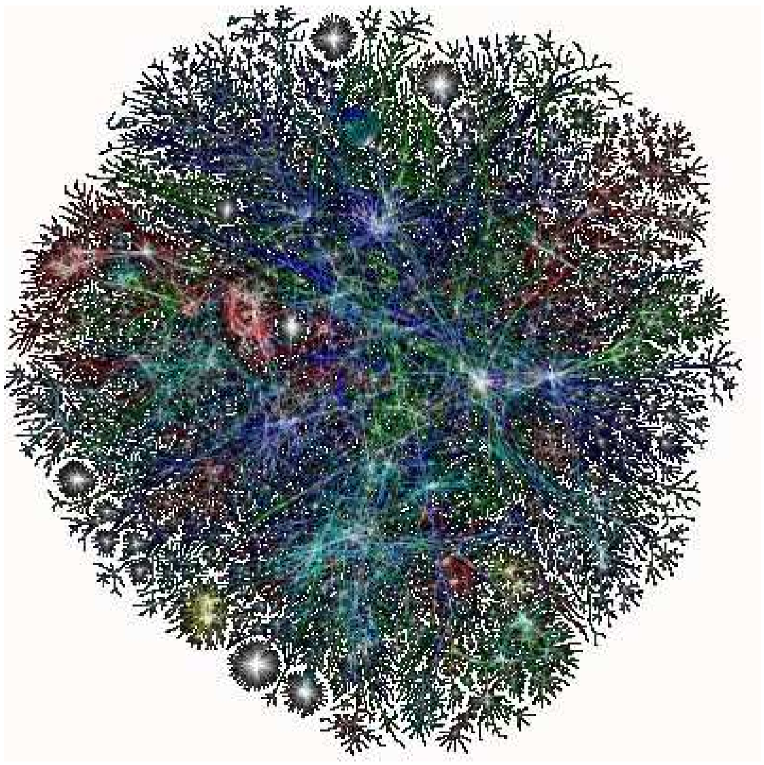

Figure 1.2: A visualization of the network structure of the Internet. Vertices represent servers, and edges represent fiber-optic links. *Image by the OPTE corporation.*

thousands of computers. In the case of small social networks, such as those studied by Moreno, visualizing them by drawing on a piece of paper is a fairly useful device, as a lot of information about the structure can be discerned upon mere examination. For example, in the social network shown in Fig. 1.1, even a cursory glance can reveal the most promiscuous individuals in the group (in this case those with the highest number of links). However, it is highly questionable, whether we can say anything about the Internet by looking at Fig. 1.2, apart from the fact that it resembles a "hairball". In fact the futility of such an approach has led some researchers to derisively term such visualizations as "ridiculograms"[3], whose meaning we think is fairly self-explanatory.

More importantly, in such large systems, asking questions such as which individual

[3]Attributed to Harvard biologist Marc Vidal. Incidentally, some have suggested that pictures such as these tend to be suitable for publications in popular journals such as Nature or Science. A set of criteria for what constitutes a ridiculogram have also been devised, however we shall not list them here.

vertex is the most central or most crucial for connectivity, is clearly meaningless in much the same way as it is (with apologies to Newtonian Determinists) to try and trace the trajectory of a single molecule in a cloud of Helium gas. This change in scale, therefore forces us to change our analytic approach, as well as the type of questions that we seek to answer. Rather than focussing on the properties of a single vertex or edge, it is more useful to look at statistical properties of networks, such as fractions of vertices that affect network connectivity.

It is largely this change of approach in studying the large-scale properties of networks that began to interest physicists, since in particular, techniques from statistical physics are well suited to be employed in such studies. Moreover, unlike social scientists who are primarily interested only in social systems, or mathematicians who study graphs as purely theoretical objects, physicists are relatively unshackled, in that their approach is largely inspired by empirical studies of a variety of real-world networks, including social, biological, technological and other systems. When talking about the large-scale properties of networks, one might ask questions such as: how do the emergent collective behaviors depend on the connectivity of the network? Physicists have thoroughly studied this question in magnetic spin systems, for which it has long been recognized that the details of a phase diagram depend on the dimensionality of the system, the symmetries of the spin-spin interactions, and the range of these interactions. Attempts by physicists to find analogs of these concepts in networks have led to the discovery of a number of previously unknown and intriguing network properties, which in turn have inspired an impressive array of new theories, techniques, algorithms, models, and measures to describe and illuminate their function.

## 1.3 Types of networks

In the beginning of this chapter, we mentioned that a network is a set of vertices connected together by edges. One can think of this as the simplest representation of a network—its most canonical form—however there are a number of extensions that we can add to reflect more complicated forms of connections. Let us consider an acquaintance network, where vertices represent people, and there is an edge between them if they are friends. One way to represent this is to use the basic representation of a network (also known as a unipartite network). However, as we all know, there are levels of acquaintance, some people are more familiar with each other than with others. An obvious way to represent this in networks is to assign weights to the edges. Taking it still further, edges do not have to represent friendship, but also animosity, so we can assign different types of edges between people. Yet another embellishment we can add is to add direction to the edges. Sometimes, we might call someone our friend, but the sentiment is not necessarily reciprocated (a common phenomena in certain high schools), so an edge can point only in one direction—if person A claims to be acquainted with person B, but not vice versa. Such networks are called *directed graphs* or *digraphs* for short. (A better example of this is an e-mail network, where vertices represent individuals and an edge represents the act of sending an e-mail, which at a given point of time clearly goes in one direction.)

Edges are not the only ones that can be assigned weights or characteristics. Vertices themselves might take on attributes that represent different categories. For example, a vertex in the acquaintance network described above can be assigned scalar characteristics such as gender, race, nationality, income level and a variety of other things. Speaking of different types of vertices, some networks can be naturally

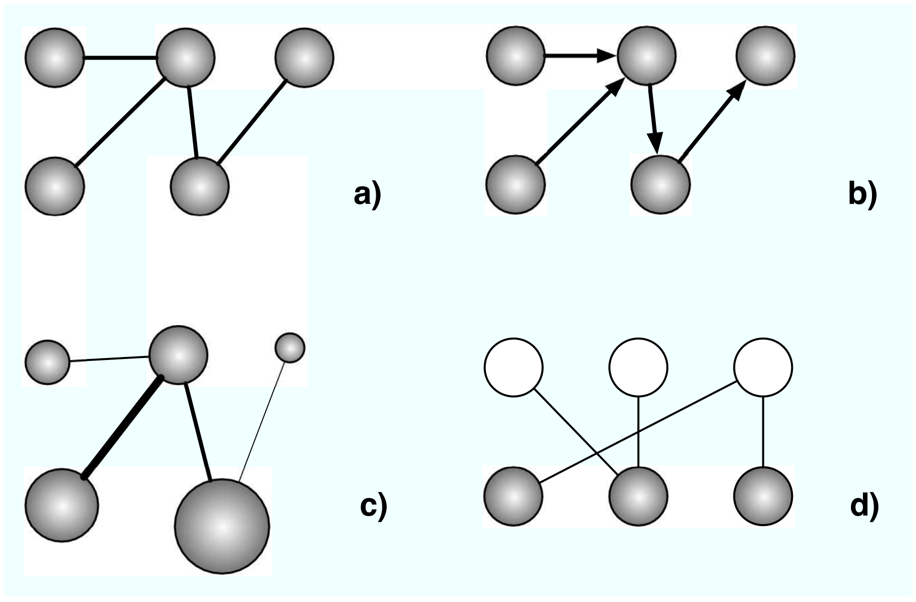

Figure 1.3: Various ways to represent networks: a) a simple unipartite undirected network; b) a directed network with edges pointing in one direction; c) a network with weighted vertices and edges representing some discrete characteristics; d) a bipartite network with two types of vertices with edges running only between unlike types.

partitioned in various ways. One example is an affiliation network, where people are joined together by common membership of some group, say a network of professionals and the companies they work for. Here, there are two types of vertices, one representing people and the other the companies, with edges running only between the two types and not within them. This is an example of a *bipartite* graph. In general, networks consisting of different types of vertices with edges running only between unlike types, are called *multipartite* networks, of which the bipartite graph is a special case.

One can also organize vertices into groups, where edges connect more than two vertices. Such graphs are called *hypergraphs* and are typically used in social networks to organize vertices into some kind of familial, topical or vocational group. In a network representing communities in a city neighborhood for example, those living in a particular house can be considered as a single unit, and a family of five can be represented as a single *hyperedge*. Hypergraphs can consist of hyperedges joining together the same or different number of vertices, or they might consist of different types of vertices, and so on.

In addition to these, there are networks that evolve in time, such that vertices or edges might appear or disappear or their characteristics (such as weights or types) might change. We will see examples of most of the networks described here, in the remainder of this dissertation, however, we hope that the reader appreciates that there are many other ways we can represent networks (indeed there surely are representations not conceived of yet). In Fig. 1.3 we show a visual representation of some of the different forms that networks can take.

## 1.4 Standard metrics and tools for measuring network structure

Having described the various types of networks, we now move on to describe some of the more common statistical and topological structural measures that are commonly used to quantify their properties.

### 1.4.1 Adjacency matrix

Perhaps, the simplest (and most common) way to represent a network is by means of the so-called *adjacency* matrix. Let us assume that there are $n$ vertices in the network, that are connected to each other in some fashion, via $m$ edges. Furthermore let these edges be undirected. Then it is possible to completely specify the connection structure of the network by an $n \times n$ matrix $\mathbf{A}$ whose elements are,

$$A_{ij} = \begin{cases} 1 & \text{if there is an edge joining vertices } i, j, \\ 0 & \text{otherwise.} \end{cases} \tag{1.1}$$

Consider for example, the simple network shown in Fig. 1.4. The adjacency matrix

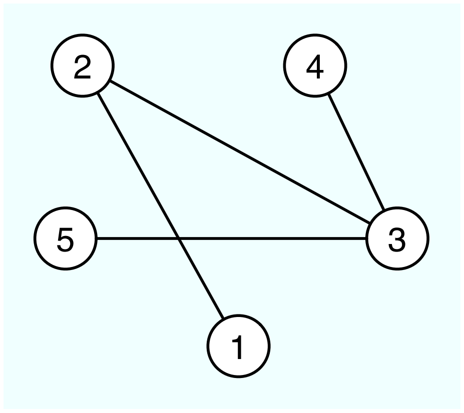

Figure 1.4: A simple undirected network consisting of five nodes and four regular edges.

representation of this network is,

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}. \tag{1.2}$$

This manner of representation is not just restricted to the case of undirected graphs, but can be extended to many different types of networks. In a directed graph, for example, the elements of the matrix are still $(0, 1)$ depending on the existence or not of an edge, however, $A_{ij} = 1$ only if there is an edge emanating from $j$ incident on $i$ and not the other way round (that is in general $A_{ij} \neq A_{ji}$). The only change in the matrix is that it might no longer be symmetric. For graphs with weights assigned to their edges, the elements of the matrix take on the values of the assigned weights, which one can write explicitly (say 2.3) or normalize them such that it takes on a value between 0 and 1. In addition, one can think of a number of embellishments and extensions to reflect other more complicated types of networks.

Despite its relative simplicity, the adjacency matrix is quite a powerful tool while analysing networks, in the sense that a lot of information is encoded in it. Reverting back to the simple example of Fig. 1.4, if one were interested in the number of edges incident on a particular vertex—conventionally referred to as the degree—one can easily compute it via the expression,

$$k_i = \sum_{j=1}^{n} A_{ij}, \tag{1.3}$$

where $k_i$ refers to the degree of vertex $i$. The total number of edges in the network can be computed by summing all elements of the adjacency matrix and from this one can calculate the mean number of edges $\langle k \rangle$ incident on a vertex,

$$\frac{1}{n}\sum_{i,j=1}^{n} A_{ij} = \frac{\sum_i k_i}{n} = \frac{2m}{n} = \langle k \rangle, \tag{1.4}$$

where the factor of 2 comes from the fact that our example network is undirected, and we are double counting the number of edges due to the symmetric nature of the matrix. An alternative statement is that we are really summing over the edge ends, and each edge clearly has two ends.

Apart from the degrees of vertices we can also calculate a number of other properties from the adjacency matrix. For example, one might be interested in finding vertices that are reachable from a given node, apart from the ones that it is directly connected to. In the network shown in Fig. 1.4, node 3 is directly connected to nodes 2, 4 and 5, but also to node 1 via node 2. A simple way to measure this is to look at the existence of *paths* in the network. In the language of graph theory, a path is a sequence of vertices such that from each vertex there exists an edge to the next vertex in the sequence. In this context the length of a path $l$ between two given vertices is just the number of edges that it traverses before it terminates at one of the two vertices. An interesting question then is, given two vertices $i$ and $j$, does there exist a path connecting the two in some fashion, and if so, are there more than one? Happily enough this can be computed in a straightforward fashion by employing the adjacency matrix. Say, we would like to find the number of paths between two vertices $i$ and $j$ that are connected by a common neighbor $k$—this is an example of all paths of length 2—then the corresponding expression for this is,

$$\sum_{k=1}^{n} A_{ik}A_{kj} = [\mathbf{A}^2]_{ij}, \tag{1.5}$$

where the sum is over all common neighbors and the last term denotes the $ij$-th element of the square of the adjacency matrix. It is relatively straightforward to convince oneself that the general expression for a path of any length $l$ between two vertices is given by taking the matrix to the power $l$ and then picking out its $ij$-th element.

Note that in the expression above, we have not imposed any restrictions on the nature of the path. The sum takes into account all possible paths even if they share a number of common vertices and edges. Sometime it is of interest to look at paths that do not share any common vertex apart from the start and end vertices. Such paths are conventionally referred to as *vertex-disjoint* or *vertex-independent* paths. A useful application of this concept is the subject of Chapter II.

Apart from the simple properties described above, one can look at more complicated aspects of network structure by examining the spectral distribution of the adjacency matrix. We desist from describing the details here, as it is not a subject dealt with in this dissertation, and instead move onto other things. The interested reader can consult [68] and the references contained therein.

### 1.4.2 Degree distribution

One of the most important and widely studied metrics of network structure is the degree distribution. As discussed, the degree $k_i$ is the number of edges connected to a vertex $i$. If, however we are to consider the network as a whole, then a more sensible measure is to look at the fraction of vertices in the network that have degree $k$. We call this the degree distribution $p_k$. Alternatively, $p_k$ represent the probability of a randomly chosen vertex to have exactly $k$ edges connected to it.

The degree distribution is useful for a number of reasons. From a practical standpoint, most mathematical models based on it are relatively straightforward and one can make precise calculations. In terms of characterizing networks in the real world, the simplest thing to measure is the degree of a vertex, and from that one can easily construct a histogram to represent the degree distribution. Perhaps most importantly, the degree distribution (usually) is an excellent indicator or guide to a network's properties both in terms of its topology as well its dynamics.

Since $p_k$ is a probability distribution it satisfies the normalization condition,

$$\sum_{k=0}^{\infty} p_k = 1. \tag{1.6}$$

From this we can calculate the various moments (of the degree) in the usual fashion. For example, the first two moments are given by,

$$\langle k \rangle = \sum_{k=0}^{\infty} k p_k, \quad \langle k^2 \rangle = \sum_{k=0}^{\infty} k^2 p_k, \tag{1.7}$$

where $\langle k \rangle$ is the average degree for the entire network, and $\langle k^2 \rangle$ is the mean squared degree.

One can also define degree distributions for more complicated forms of networks. In a bipartite graph which consists of two different types of vertices, with edges running only between unlike types, there is a separate degree distribution for each type of vertex. Consider for example, an affiliation network of $M$ movies and $N$ actors where each actor has appeared in an average of $\mu$ movies and each movie has an average cast of $\nu$ actors. Suppose that there are $m$ edges in the network, then the first thing to note is that $\mu$ and $\nu$ are fixed by the condition,

$$N\mu = M\nu = m, \tag{1.8}$$

since each edge joins together exactly one movie and one actor. We then define $p_j$ to be the fraction of actors who have appeared in $j$ movies, and $q_k$ to be the fraction of movies that have had a cast of $k$ actors. These then satisfy the sum rules

$$\sum_{j=0}^{\infty} p_j = \sum_{k=0}^{\infty} q_k = 1, \tag{1.9}$$

and,

$$\sum_{j=0}^{\infty} j p_j = \mu, \quad \sum_{k=0}^{\infty} k q_k = \nu. \tag{1.10}$$

For directed networks, on the other hand, each vertex has separate in-degree and out-degree for links running in and out of that vertex. The degree distribution in this case is a function of two variables $p_{jk}$ representing the fraction of vertices that simultaneously have in-degree $j$ and out-degree $k$. Since every edge in a directed graph must leave some vertex and enter another, the net average number of edges entering a vertex is zero, and $p_{jk}$ is thus constrained by the condition,

$$\sum_{jk} (j-k) p_{jk} = 0. \tag{1.11}$$

In a similar fashion one can define a degree distribution for a variety of other cases. In Chapter III we present a model where we extend the concept of degree distributions to hypergraphs.

In the real world the degree distribution comes in a variety of flavors such as exponential and Gaussian, although these are relatively rare. By far the most common type is where vertices have degrees that are highly right-skewed, implying that their distribution has a long right-tail of values that are far above the mean. Typically (but not always) these take on the form of power-laws,

$$p_k \sim k^{-\alpha}, \tag{1.12}$$

where $\alpha$ (in most cases) takes on a value between 2 and 3. Although as mentioned before, one can just construct a histogram of the degrees to represent its distribution, measuring the tail is particularly cumbersome, as in most cases one does not have an accurate sample of measurements to get good statistics.

There are usually two methods to get around this problem. The first method is to progressively increase the bin size (usually exponentially with the degree). The simplest way to do this is to double it at intervals, for example, a series of bins might look like, 1, 2-3, 4-7, 8-15 and so on. The number of samples in each bin is then divided by its width to normalize the measurement. If the histogram is plotted on a logarithmic scale, the widths of the bin will appear even. As we get near the tail, the bins get much wider and thus the problems with the statistics are reduced. A significant drawback with this method, however, is that any differences between values that fall in the same bin are lost.

A much cleaner way to present the degree data that avoids these problems is to make a plot of the cumulative distribution,

$$P_k = \sum_{k'=k}^{\infty} p'_k, \tag{1.13}$$

which is the probability of a vertex to have degree greater than or equal to $k$. This is also known as a rank frequency plot, since in effect it ranks the fraction of vertices with degree $k$ in decreasing order of their occurrence. Plotting the distribution in this fashion significantly reduces the noise in the tail as all data points are included.

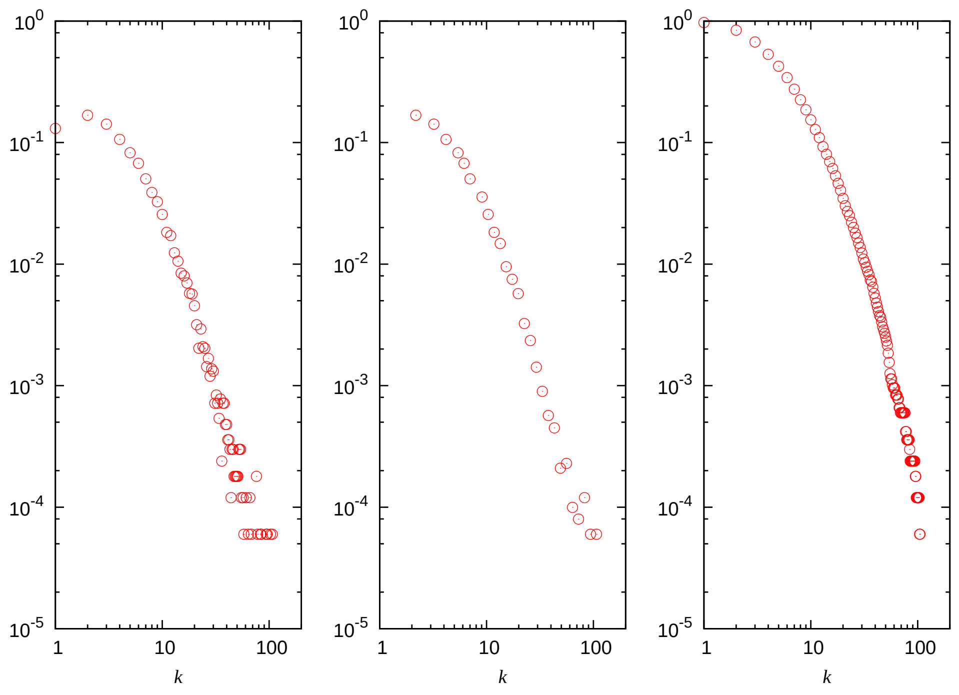

Figure 1.5: Three different ways to plot the degree distribution as described in the text. From left to right, the degree distribution plotted via linear binning, logarithmic binning and finally the plot of the cumulative distribution. The data is taken from a collaboration network of condensed matter physicists [106].

To illustrate this, in Fig. 1.5 we show three different ways to plot the degree distribution. The data in the plot is taken from a collaboration network of condensed matter physicists [106], where the vertices in the network represent physicists, and there is an edge between two of them if they have written a paper together. From the figure, one can see the advantages and drawbacks of each of these methods. The leftmost plot is just a linear binning of the degree distribution plotted on a log-log scale. Towards the tail of the distribution, the plot tends to get messy due to the scarcity of sample data points in that region. The central plot is an example where the data is binned logarithmically and as the figure clearly shows, the noise in the tail is significantly reduced, however at the cost of making the bin sizes much larger and thus throwing out (potentially) valuable information. In the rightmost plot, we show the cumulative distribution of the degrees that shares none of the disadvantages of the previous plots—all data points are included, and the tail is relatively clean.

### 1.4.3 Transitivity

In many real networks—particularly social networks—it is found that if a vertex 1 is connected to vertex 2 and vertex 3, then it is very likely that vertices 2 and 3 are also connected to each other. In the context of acquaintance networks, this is equivalent to saying that the friend of my friend is quite likely my friend as well. This property is conventionally referred to in the literature as *transitivity.* In terms of network topology, transitivity is the measure of the presence of triangles in the network—sets of three vertices that are all connected to each other. This property can be quantified by means of a coefficient (also referred to as the clustering coefficient) defined in the following manner,

$$C = \frac{3n_{\triangle}}{n_{triples}}, \tag{1.14}$$

where $n_{\triangle}$ refers to the number of triangles in the network and $n_{triple}$ refers to the number of connected triples—a single vertex with edges running to a pair of other vertices. The factor of 3 in the numerator comes from the fact that each triangle contributes three triples, and thus ensures that $C$ lies in the range $0 \leq C \leq 1$. Essentially the clustering coefficient $C$ measures the fraction of triples, that are closed by the presence of a third edge thus converting it into a triangle, however it might be easier to think of it in terms of the mean probability of two neighbors to share a common third neighbor.

There is an alternative definition of the clustering coefficient, first proposed by Watts and Strogatz [149], who considered a local measure $C_i$ for a vertex $i$ defined in the following manner,

$$C_i = \frac{n^i_{\triangle}}{n^i_{triples}}, \tag{1.15}$$

where the numerator represents the number of triangles connected to vertex $i$, and the denominator refers to the the number of triples centered on $i$. The overall clustering coefficient for the network is then simply,

$$C = \frac{1}{n}\sum_i C_i. \tag{1.16}$$

Note, that Eqns (1.14) and (1.16) give different results for the overall value of $C$. In the former case, we take the ratio of the mean values of $n_{\triangle}$ and $n_{triple}$ whereas in the latter case we compute the mean of the ratio of these two values. The main difference between the two definitions is that Eqn. (1.16) assigns a higher weight to connected triplets incident on lower degree nodes. Nevertheless both definitions of the clustering coefficient are in wide use in the literature, and the choice of one or the other is a function of the type of question that researchers are trying to answer.

It is worth noting that interest in the phenomenon of clustering has been inspired mostly by empirical measurements on real networks, where this is fairly commonplace. Networks generated by standard mathematical models (mostly random graph models, described in detail in the next section), on the other hand, do not have a high degree of clustering. In fact it a well known result that on the random graph $C = \mathrm{O}(n^{-1})$, which is to say for large graphs sizes, as $n \to \infty$ (the standard limit in the study of theoretical graph models, also known as the thermodynamic limit in statistical physics), this is a vanishing quantity. In the real world, in contrast, one expects $C$ to tend to a non-zero limit as the network gets larger.

Most attempts to reproduce finite clustering in theoretical network models have fallen short, see [142, 31, 123], however, recently Newman [112] has proposed a random graph model that seems to reproduce clustering in a reasonable fashion. In particular one can specify the level of clustering in the graph *a priori* and then have the model reproduce it in the networks that it generates.

The clustering coefficient is a measure of the presence and density of triangles in the network, which is just a closed loop of length three. An obvious generalization of this concept is to look at loops of longer length. There have been some studies in this direction [23, 33, 62], however as of this writing there is no complete theory of such structures.

### 1.4.4 Assortativity and degree correlations

As mentioned earlier in the text, vertices in a network can be endowed with certain extrinsic characteristics. Consider for example, a network of academics in a university setting, vertices here represent Professors or researchers, and edges might represent some manner of acquaintance. An obvious way to distinguish between vertices in this case, is if there is some scalar attribute which represents academic discipline (physics, mathematics, biology, political science etc.). Another example is provided by networks in a food web. Vertices in this case represent organisms, and edges run between them if one organism is a source of nourishment for the other. One can then classify vertices in terms of trophic levels, say plants, herbivores, omnivores, carnivores and so on.

In these types of networks, an obvious question to ask, is whether vertices of a certain type connect only to vertices of a similar type, or to unlike types, or are connections distributed in some fashion between multiple types? In an academic acquaintance network, two physicists are more likely to know each other than someone who is a political scientist. In the food web, there are a number of edges connecting plants and herbivores, many more that link herbivores and carnivores, but not too many that link a plant to another plant, or a herbivore to a herbivore. This type of selective linking is conventionally referred to as *assortative mixing* or *homophily*. It is also occasionally referred to as *assortative matching* in the ecology literature.

This phenomenon of assortative mixing can be quantified by means of an assortative coefficient [107]. Let $E_{ij}$ be the number of edges in the network that connect a vertex of type $i$ to one of type $j$, with $i, j = 1, \ldots, n$, then similar in spirit to the adjacency matrix for vertices, we can represent these edges in the form of an edge incidence matrix $\mathbf{E}$, with elements $E_{ij}$. We then define a normalized mixing matrix,

$$\mathbf{e} = \frac{\mathbf{E}}{||\mathbf{E}||}, \tag{1.17}$$

where $||\mathbf{E}||$ refers to the sum of the elements of the matrix $\mathbf{E}$. The entries $e_{ij}$ in the normalized matrix represent the fraction of edges that connect vertices of types $i$ and $j$, and satisfies the normalization condition,

$$\sum_{ij} e_{ij} = 1. \tag{1.18}$$

The assortativity coefficient $r$ is then defined thus,

$$r = \frac{Tr(\mathbf{e}) - ||\mathbf{e}||^2}{1 - ||\mathbf{e}||^2}, \tag{1.19}$$

where $Tr(\mathbf{e})$ is the standard matrix trace—the sum of the diagonal elements $e_{ii}$. The value of the coefficient $r$ lies in the range $-1 \leq r \leq 1$, where 1 represents a perfectly assortative network, 0 a randomly mixed one and -1 a perfectly disassortative network. Note that the definition of $r$ in the equation above is closely related to the Pearson correlation coefficient widely used in Statistics [129].

A special case of assortative mixing based on the degrees of vertices is referred to as degree correlation. In this case, we are interested in the question: Do high-degree vertices in a network preferentially associate themselves with other high-degree vertices or do they instead connect to low-degree ones? Reverting back to our example of friendship networks, this is similar to asking whether popular people tend to associate with each other, or otherwise. Since the degree is an important topological measure, degree correlations assume a significant amount of relevance as they can give rise to complicated network structural effects. The degree correlation can be computed using Eqn. (1.19), where the elements $e_{ij}$ represent the fraction of edges that connect a vertex of degree $i$ to that with degree $j$.

The concept of assortativity can also be generalized to more complicated structures such as directed networks, we refer the interested reader to [107] for further details.

### 1.4.5 Centrality

A question of practical importance, that has been considered by many researchers in the field of networks and graph theory, is to determine the relative importance of a vertex within a graph. For example, in a social network, we might want to determine who is the most important person, or in a information network like the World Wide Web, try and identify the most popular web-page. The amount of importance of a particular vertex is conventionally referred to as its *centrality.* There are a number of ways of defining centrality depending on the type of network, and the type of question we seek to answer.

The simplest measure of centrality is the *degree centrality.* The degree centrality is defined as the number of links incident on a node, so the higher the degree of the node the more central it is. This is a fairly straightforward measure, and is easy to understand in the context of the popular search engine Google, which, by virtue of popularity of its usage has an extensive fraction of hyperlinks to other web-pages. It is thus considered highly central in the sense considered here. Mathematically the degree centrality of a vertex $i$ is written as,

$$c_d(i) = \frac{k_i}{n-1}, \tag{1.20}$$

where $k_i$ is the degree of vertex $i$ and $n$ is the number of nodes in the network.

Having a large number of connections to other vertices is not the only way for a node to be central. For example, in the corporate world it is common wisdom that the road to success is not a function of how many people we know, but more importantly it's who we know (something that is achieved by a process coincidentally referred to as networking). In this case importance or centrality of a particular vertex is a function of the centralities of all of its neighbors. If a node is connected to highly central neighbors, then the more central it is itself. This definition of centrality is referred to as the *eigenvector centrality* [25]. The centrality of a vertex $i$, say $x_i$ can be written as

$$x_i = \lambda^{-1} \sum_{j=1}^{n} A_{ij} x_j, \tag{1.21}$$

where $\lambda$ is some constant and $A_{ij}$ is the adjacency matrix. The equation essentially states that the centrality of $i$ is equal to the sum of the centralities of all its neighbors upto a multiplicative constant. The equation can be recast in the following matrix form,

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{x}, \tag{1.22}$$

which is just the standard eigenvalue equation, with $\mathbf{x}$ an eigenvector of $\mathbf{A}$. The requirement that all entries of the eigenvector $\mathbf{x}$ be positive implies that the one corresponding to the leading or largest eigenvalue $\lambda$ gives us the correct centrality measure (this follows from the Perron-Frobenius theorem, a well-known result in the field of linear algebra). Once this eigenvector is determined, the centrality of a vertex $i$ is given by the $i$th component of the eigenvector. A practical way to find this eigenvector is to use the multiplication method, where we guess an initial eigenvector—the unit vector with all ones as its components for example—and then repeatedly multiply it into Eqn. (1.22) until it converges to a fixed value for $\mathbf{x}$. Note that Google employs a variant of this measure called PageRank [27] to rank web-pages in order of their importance.

In some networks vertices are central in that they have a large number of connections (high degree centrality), but nevertheless occupy a peripheral position in the network in terms of having a large average distance to other vertices. Consequently yet another way to define the centrality is related to the concept of path lengths, that we talked about in an earlier section. The definition of centrality in this case, called the *closeness centrality*, is related to the distance between vertices in terms of the sequence of edges connecting them [133]. Vertices are considered to be close to each other in the sense of the shortest possible path connecting them. Let $d_{ij}$ denote the shortest possible path connecting two vertices $i$ and $j$—also referred to as the geodesic—then the closeness centrality of a vertex $i$ is defined as,

$$c_c(i) = \frac{n-1}{\sum_{j \neq i} d_{ij}}, \tag{1.23}$$

where the condition $j \neq i$ ensures that the sum does not diverge (since $i$ is at a distance of zero from itself). Equation (1.23) measures the reciprocal of the average geodesic distance of a vertex to all others reachable from it, thus the smaller the distance, the larger the contribution to the centrality. In Chapter VI we discuss a game theoretical model on networks based on the concept of the closeness centrality.

Apart from the measures described here, there are a number of other ways to define the centrality, such as the *Katz centrality, betweenness centrality, information centrality* etc.—see for instance [86, 61, 141].

### 1.4.6 Components

A particularly useful measure of network structure (and a subject dealt with extensively in this dissertation) is the connected component of a graph. In the context of an undirected graph a connected component is defined as a set of vertices in the the network that are all linked to each other such that there exists some path between any two pairs in the set. The network shown in Fig. 1.4 constitutes a connected component as per the definition considered here.

The notion of a component assumes particular significance while studying the spread of epidemics on a network. For example, we might represent a village or a town as a network where the nodes represent people, and edges represent acquaintance. In such a setting, starting from an infected individual we can study the spread and contagion of diseases such as the flu virus, or malaria. If the source of an infectious disease is an individual who is part of the network consisting of only a few people then it's quite likely that it will not cause much damage and die out fairly soon. If however the infected individual is part of a group of people that encompasses an extensive fraction of the network, then this could lead to dire consequences, as the disease has the potential to spread to the entire town. This brings us to the idea of the *giant component*. A giant component of a graph is defined as the set of vertices of size $\mathrm{O}(n)$—$n$ being the number of nodes in the network—such that any two in the set are reachable by some path. A large amount of research has been devoted to the study of the giant component and its properties in both real and model networks. The majority of studies have focussed on two aspects. The first is to determine whether a network has a giant component to begin with, and if so measure its size. The second is to examine the effects of node and edge removal (in some fashion) on the size of the giant component—this is similar to simulating epidemic

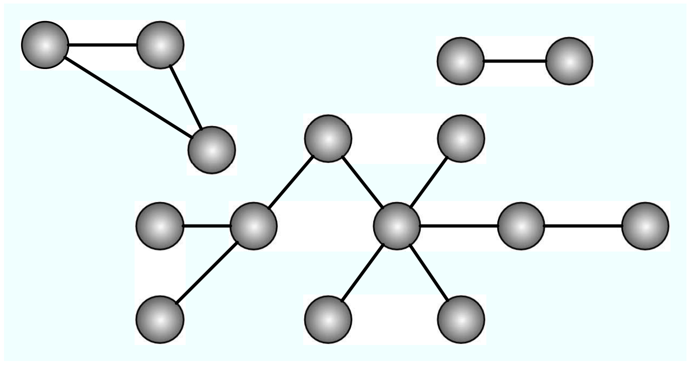

Figure 1.6: A network with three connected components. The sets of two and three vertices are referred to as small components, while the larger set is the giant component.

spreading on a network, or the effect of disruptions on a communication network. The latter is closely related to percolation theory in physics, and in that sense the giant component is equivalent to the infinite or spanning cluster in percolation, though strictly speaking the comparison is valid only in the thermodynamic limit.

Most networks, both real and model, are found to have at most one giant component, and a number of other connected components of size O(1), conventionally referred to as small components. As an example of this, Fig. 1.6 shows a network consisting of a single giant component of ten nodes and two small components of size three and two.

Just as in the case of other properties, component structure can be extended to more complicated types of networks. For directed graphs there are two types of components, *strongly connected* and *weakly connected*. The definition of the component is the same as before, however we need to account for the direction of the edges. Thus a strongly connected component in a directed graph is defined as the set of vertices in the network, such that for any pair $i$ and $j$, there exists a path from $i$ to $j$ *and* from $j$ to $i$, which in a directed graph is not necessarily the case. If we relax this condition, then we have a weakly connected component, which is just the analog for a component in an undirected graph.

For multipartite graphs, we can have components for each type of vertex, or components consisting of multiple types and so on. In Chapter III we describe in detail the component structure of a tripartite graph consisting of three types of vertices.

### 1.4.7 Generating functions

A significant part of the mathematical material presented in this dissertation is treated with the technique of generating functions. Before we move on, it is worth providing a brief primer on the use of this method.

In mathematical terms a generating function is a formal power series whose coefficients contain information about a sequence, say $a_k$, where $k$ is a natural number. There are various types of generating functions depending on the type of sequence, for example Exponential and Dirichlet generating functions have found wide use in certain branches of physics, however the type we consider here is the most basic type, known as the ordinary generating function.

The most fundamental generating function that we will use in our analysis is one for the degree distribution $p_k$. Let us consider a unipartite undirected graph consisting of $n$ vertices with $n$ large, then we define,

$$G_0(z) = \sum_{k=0}^{\infty} p_k z^k. \tag{1.24}$$

Since $p_k$ is a probability, the distribution is assumed normalized, i.e $\sum_k p_k = 1$. Thus we have,

$$G_0(1) = 1. \tag{1.25}$$

The coefficients of $G_0(z)$ represent probabilities and the function is therefore called a probability generating function. As the probability distribution is normalized and positive-definite, $G_0(z)$ is absolutely convergent for all $|z| \leq 1$ and therefore has no singularities in this region. All the calculations in which we will employ generating functions are restricted to the region $|z| \leq 1$.

There are a number of properties of generating functions that make them a particularly powerful tool in performing analytical calculations. For example, the probability $p_k$ for any $k$ can be found simply by taking the $k$th derivative of $G_0(z)$,

$$p_k = \frac{1}{k!} \frac{\mathrm{d}^k G_0}{\mathrm{d}x^k}\bigg|_{z=0}, \tag{1.26}$$

from which we can see that $G_0(z)$ contains all the information about $p_k$, or in other words $G_0(z)$ *generates* the distribution $p_k$. Calculating the moments of the distribution is also a fairly simple exercise. The average degree $\langle k \rangle$ is given by,

$$\langle k \rangle = \sum_k k p_k = G_0'(1). \tag{1.27}$$

Higher moments can be calculated by taking larger derivatives of the function, and in general,

$$\langle k^s \rangle = \sum_k k^s p_k = \left[ \left( z \frac{\mathrm{d}}{\mathrm{d}z} \right)^s G_0(z) \right]_{z=1}. \tag{1.28}$$

An important property of generating functions is the *powers* property. Say $k$ represents a property of an object (degree for instance) whose distribution is generated by some generating function. The distribution of the total of $k$ summed over $m$ multiple independent realizations is generated by the $m$'th power of the generating function. For example, if we pick two vertices at random from a network, then the distribution of the sum of the two degrees is given by,

$$\begin{aligned}
[G_0(z)]^2 &= \left[ \sum_k p_k z^k \right]^2 = \sum_{jk} p_j p_k z^{j+k} \\
&= p_0 p_0 z^0 + (p_0 p_1 + p_1 p_0) z^1 + (p_0 p_2 + p_1 p_1 + p_2 p_0) z^2 \\
&\quad + (p_0 p_3 + p_1 p_2 + p_2 p_1 + p_3 p_0) z^3 + \ldots
\end{aligned} \tag{1.29}$$

As can be seen clearly from the equation, each coefficient of the power of $z^s$ is the sum of all the possible combinations of the product $p_j p_k$ such that $j + k = s$, which gives us the correct expression for the probability of the two vertices to have a combined degree equal to $s$. It is straightforward to see that the corresponding expression for an arbitrary number of vertices is just $[G_0]^m$.

Apart from $G_0(z)$, there is another type of generating function that we will be using extensively. In certain situations—primarily in calculating properties of networks generated by the configuration model (discussed in Sec 1.5.2)—it is of interest to us to look at the distribution of edges of a vertex that we arrive at by following a randomly chosen edge, which in general is not the same as $p_k$. The edge arrives at a vertex with probability proportional to the degree of that vertex, and therefore the distribution of edges of that vertex is proportional to $kp_k$. The correctly normalized distribution is generated by,

$$\frac{\sum_k kp_k z^k}{\sum_k kp_k} = z\frac{G_0'(z)}{G_0(1)}. \tag{1.30}$$

Thus if we choose a vertex with degree $k$ at random and follow each of its $k$ edges to its neighbors, then the vertices arrived at each have the distribution of outgoing remaining edges generated by this function, less one power of $z$ to account for the edge that we arrived along,

$$G_1(z) = \frac{G_0'(z)}{G_0(1)} = \frac{1}{\langle k \rangle} G_0'(z). \tag{1.31}$$

The outgoing edges of a vertex are also referred to as the *excess degree* of the vertex and the probability of a vertex to have excess degree $k$ is denoted $q_k$, where,

$$q_k = \frac{(k+1)p_{k+1}}{\langle k \rangle}, \tag{1.32}$$

thus we can also write $G_1(z)$ more compactly as,

$$G_1(z) = \sum_k q_k z^k. \tag{1.33}$$

Taken together $G_0(z)$ and $G_1(z)$ are the fundamental generating functions for the network, and in most cases it is these that we will use to calculate the properties of interest. When appropriate we will define more complex forms of generating functions, however the underlying concepts and principles are identical to that discussed here.

There are of course a number of other important properties and measures of networks such as community structure, modularity, motif counts and so on. These are however beyond the scope of this dissertation and we therefore have restricted ourselves to what is relevant to the material to follow. For those with an interest in material other than covered in this dissertation, we recommend they have a look at [53, 113, 108], which are all excellent reviews of major developments in the field.

## 1.5 Random Graphs

Having defined some of the basic structural quantities and analytical tools used to describe networks, we are now in a position to formulate mathematical models to describe network properties. One of the most important models are those that fall in the category of *random graphs.*

### 1.5.1 Poisson random graphs

The simplest (and perhaps most influential) model for a network was proposed by Solomonoff and Rapaport [139] and independently by Erdős and Rényi [56]. If there are $n$ vertices in a network, then with some fixed probability $p$ we connect each pair (or not) with an edge. This is what is known as the $G_{n,p}$ model—since the main parameters are the number of vertices $n$ and the connection probability $p$. The $G_{n,p}$ model is the ensemble of all graphs in which a graph with $m$ edges occurs with probability $p^m(1-p)^{M-m}$, where $M = \frac{1}{2}n(n-1)$ is the maximum number of possible edges in a network of $n$ nodes. A related model defined by the same authors, called $G_{n,m}$, is the ensemble of all graphs having $n$ vertices and $m$ edges where each possible graph occurs with equal probability. Both these models are equivalent to the *canonical* and *grand-canonical* ensembles in statistical physics. For example the probability $p$ plays the role of a *field* and $m$ is equivalent to an *order parameter*. One can define the analog of the Gibbs and Helmholtz free energies, which in this case correspond to generating functions for moments of graph properties.

The model as described above is exactly solvable for a number of properties in the limit of large graph size—as shown by Erdős and Rényi in a series of famous papers [56, 57, 58]. The degree distribution for the model is binomially distributed since each edge occurs with independent fixed probability and therefore,

$$p_k = \binom{n}{k} p^k (1-p)^{n-k}. \tag{1.34}$$

The mean degree of the graph, say $\mu$ is just $\mu = (n-1)p$. If we take the limit $n \to \infty$ while keeping the average degree fixed, then Eqn. (1.34) tends to a Poisson distribution,

$$p_k = \frac{\mu^k e^{-\mu}}{k!}. \tag{1.35}$$

Consequently the model is also referred to as the Poisson random graph.

The structure of the networks generated by the model clearly depend on the value of $p$. For low values of $p$ there will be very few edges between vertices, whereas on the other end of the spectrum with $p$ close to one we will get networks where almost all vertices are connected to each other. Intuitively then, one expects to find some transition between these two states and in fact that is precisely what the model predicts. Both sets of authors have demonstrated that there exists a phase transition between a low density, low $p$ state, where vertices are connected together in small components whose sizes are exponentially distributed with a finite mean, to a high density, high $p$ state where an extensive fraction of vertices (size $\mathrm{O}(n)$) clump together to form a giant component with the remainder of vertices still connected together in small components.

Using a simple argument, we can calculate the expected size of the giant component for networks generated by the model. (This is not how Erdős and Rényi derived it, but it has the virtue of simplicity and gives the same result.) Let $u$ be the probability that we pick a vertex at random and that it does *not* belong to the giant component. Equivalently $u$ represents the fraction of vertices in the network that are not part of the giant component. In order for the vertex to not belong to the giant component, it must be that none of its neighbors are part of it either, which happens with probability $u^k$ for a vertex with degree $k$. Averaging over the distribution in Eqn. (1.35) we get the following self-consistency relation,

$$u = \sum_k p_k u^k = \mathrm{e}^{-\mu} \sum_k \frac{(\mu u)^k}{k!} = \mathrm{e}^{\mu(u-1)}. \tag{1.36}$$

The fraction of vertices that are part of the giant component (which we will denote $S$) is $S = 1 - u$ and therefore, we get the expression,

$$S = 1 - \mathrm{e}^{\mu(u-1)} = 1 - \mathrm{e}^{-\mu S}. \tag{1.37}$$

There is no closed-form solution for Eqn. (1.36). The standard way to solve for these types of transcendental equations is to guess an initial solution for $u$, say $u = \frac{1}{2}$ and then repeatedly iterate the equation until it converges to a fixed value. Intuitively though it is easy to see that the only non-negative solutions to Eqn. (1.36) are $S = 0$

for $\mu < 1$ and another solution for $\mu > 1$ that corresponds to the expected size of the giant component. The phase transition takes place at $\mu = 1$. It is not too hard to show this explicitly; in order for there to be a giant component, there must be a solution for $u$ that is less than 1, otherwise by definition of $u$ there are no vertices in the giant component. Let it take on a value slightly less than 1 thus,

$$u = 1 - \epsilon, \tag{1.38}$$

Substituting this into Eqn. (1.36) and performing a Taylor expansion we get,

$$\begin{aligned} 1 - \epsilon &= 1 - \mu\epsilon + \mathrm{O}(\epsilon^2) \\ \epsilon &= \mu\epsilon + \mathrm{O}(\epsilon^2), \end{aligned} \tag{1.39}$$

which tells us that the condition for there to be a giant component is $\mu \geq 1$. It can also be shown that the mean size $\langle s \rangle$ of a cluster to which a randomly chosen node belongs to is given by the expression,

$$\langle s \rangle = \frac{1}{1 - \mu + \mu S}, \tag{1.40}$$

which as can be seen diverges at the point $\mu = 1$. In the language of the theory of phase transitions, $S$ functions as an order parameter, and $\langle s \rangle$ plays the role of the order-parameter fluctuations. Consequently there are a series of critical exponents that describe the behavior of the system at and near the phase transition, and all of these are known. In addition to this a variety of other results are known, such as the distribution of sizes of small components above and below the transition, the mean path length between vertices and many more.

Unfortunately the properties of the random graph do not correspond well to that found in real world networks. To begin with, degree distributions in real networks, are very far from resembling a Poisson distribution. Properties such as clustering, assortativity, and correlations between degrees are also virtually non-existent. Nevertheless, from a pedagogical point of view the random graph still retains its importance. Even though it is not exactly an accurate representation of real networks, it qualitatively captures some of the basic features. When one talks about the existence of a giant component and a number of smaller components, a phase transition from a low density to high density state—features found in most real and sophisticated model networks—these ideas have all been inspired by the study of the random graph.

### 1.5.2 Generalized random graphs

There are a number of ways in which we can extend random graph models to better reflect properties of networks in the real world. An obvious improvement is to incorporate more realistic degree distributions, and this leads us to consider the *configuration model*.

In the configuration model we specify a degree distribution $p_k$ *a priori*. From this distribution, we then choose a set of $n$ values of degrees $k_i$, with $i = 1, \ldots, n$ ($n$ being the number of vertices). This is equivalent to assigning each vertex $i$, $k_i$ stubs or edge-ends chosen from this sequence. Having done this, we then choose pairs of stubs at random and connect them together. The process generates all possible configurations of graphs (hence the name configuration model) with the specified degree sequence, each instance or configuration occurring with equal probability [102]. The networks generated by the model are thus drawn from the ensemble of all graphs with a given degree sequence, with each realization having equal weight.

A variety of precise results are know about the configuration model—see [9, 21, 38, 39, 101, 102, 117] for instance. We will give a brief description of some of the more important results related to components, using the generating function formalism that we described in Sec. 1.4.7. (First introduced by Newman *et al.* [117].)

In the model the probability of an edge occurring between two vertices $i$ and $j$ is given by the expression,

$$P_{ij} = \frac{k_i k_j}{2m}, \tag{1.41}$$

where $k_i$ and $k_j$ are the degrees of vertices $i, j$ and $m$ is the total number of edges. As in the case of the Poisson random graph, calculations are made in the large $n$ limit. If we let $n \to \infty$, while keeping the average degree $\langle k \rangle = 2m/n$ fixed, we see that the number of edges $m$ is of $\mathrm{O}(n)$—networks of this type are called *sparse graphs.* In this limit, the probability of an edge occurring between two vertices $i$ and $j$ is proportional to $n^{-1}$. Among other things, one of its implications is that the graph is locally tree-like, or in other words, the probability of finding a closed loop of vertices in a small component of the graph goes as $n^{-1}$. This is crucial in terms of finding solutions to the configuration model.

Keeping this point in mind, we define two generating functions, one for the regular degree $p_k$ and one for the excess degree $q_k$, that we discussed in Sec. 1.4.7 thus,

$$G_0(z) = \sum_{k=0}^{\infty} p_k z^k, \quad G_1(z) = \sum_{k=0}^{\infty} q_k z^k. \tag{1.42}$$

Using these we can for instance, calculate the distribution of sizes of connected components in the graphs. Let $H_1(z)$ be the generating function for the size distribution of components—excluding the giant component—that are reachable by following a randomly chosen edge and then following it to one of its ends. Exploiting the local tree-like property of the graph we can conclude that the components themselves are tree-like in structure consisting of the single site we reach by following an initial edge, including any number of other tree-like clusters, with the same size distribution con-

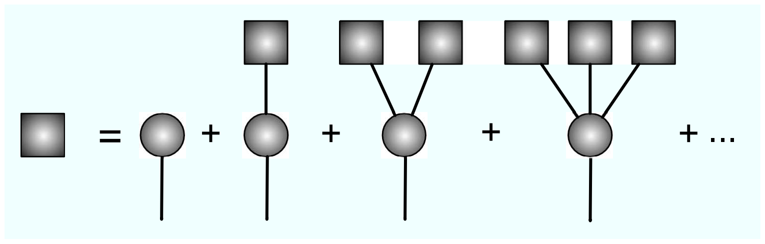

Figure 1.7: Visual representation of the sum rule for connected components of vertices reached by following a randomly selected edge as per Eqn (1.43). Each component (squares) can be expressed as the sum of the probabilities of a having a single vertex (circles), a single vertex connected to one component, to two components and so on.

nected to it by single edges. Using the powers property discussed in Sec. 1.4.7, the function $H_1(z)$ must satisfy the following self-consistency relation,

$$H_1(z) = zq_0 + zq_1H_1(z) + zq_2[H_1(z)]^2 + \dots \tag{1.43}$$

A visual representation of this is shown in Fig. 1.7. This is however just the function $G_1(z)$ evaluated at $H_1(z)$ with an overall multiplicative factor of $z$. We can thus re-write it as,

$$H_1(z) = zG_1(H_1(z)). \tag{1.44}$$

If we instead pick a vertex at random then the corresponding distribution is,

$$H_0(z) = zG_0(H_1(z)), \tag{1.45}$$

as there is a component at the end of each edge connected to the vertex apart from itself.

All information about the component sizes is encoded in the functions $H_0(z)$ and $H_1(z)$. These equations are typically transcendental in nature and it's usually hard to find closed-form solutions, however recently specialized analytical and numerical techniques have been devised to solve for them [111]. Using these methods one can, for example, extract the coefficients of $H_0(z)$, which tells us the probability of a randomly chosen vertex to belong to a component of a particular size.

The average properties of clusters can be found in a straightforward fashion. The mean size of a cluster below the phase transition (no giant component in the graph) is given by,

$$\langle s \rangle = H_0'(1) = 1 + \frac{G_0'(1)}{1 - G_1'(1)}. \tag{1.46}$$

This diverges at the point $G_1'(1) = 1$, which marks the phase transition at which a giant component first appears.

Above the transition there is a giant component that occupies a fraction $S$ of the graph. The size of the giant component can be calculated in similar fashion to the method we used for the Poisson random graphs with some subtle differences. Let $u$ be the probability that a randomly chosen *edge*—in the previous case it was a vertex, however for a Poisson distribution the result is the same regardless—leads to a vertex that is not part of the giant component, which is possible only if none of its remaining edges lead to the giant component. These edges are distributed according to $q_k$, and therefore $u$ must satisfy the self-consistency condition,

$$u = \sum_k q_k u^k = G_1(u), \tag{1.47}$$

where $k$ is the excess-degree. The probability of randomly choosing a vertex that is not part of the giant component is $G_0(u)$, and therefore $S$ is given by,

$$S = 1 - G_0(u), \tag{1.48}$$

with $u$ the solution to Eqn. (1.47).

As an example of the application of the results, let us consider a network with links distributed exponentially, such that,

$$p_k = (1 - \mathrm{e}^{-\lambda})\mathrm{e}^{-\lambda k}, \tag{1.49}$$

where $\lambda$ is some constant. Substituting this into Eqn. (1.42), we get,

$$G_0(z) = \frac{e^\lambda - 1}{e^\lambda - z}, \quad G_1(z) = \left(\frac{e^\lambda - 1}{e^\lambda - z}\right)^2. \tag{1.50}$$

The phase transition occurs at the point $G_1'(1) = 1$ which for our example network can be written as,

$$2 = e^\lambda - 1, \tag{1.51}$$

which gives us a critical value $\lambda_c = 1.09861\ldots$ In other words, a giant component exists only if $\lambda < \lambda_c$. The value of $u$ in Eqn. (1.47) is a solution of the equation,

$$u = \left(\frac{e^\lambda - 1}{e^\lambda - u}\right)^2, \tag{1.52}$$

which is a cubic equation in $u$ and can be solved analytically—the exponential distribution is one of the few cases that has an analytic solution—after which we substitute it into Eqn. (1.48) to get the size of the giant components $S$. For example, if $\lambda = 0.5$, the corresponding values are $u \approx 0.2$ and $S \approx 0.55$.

The configuration model can be extended to reflect even more complicated structures such as directed and bipartite graphs [117]. In Chapter III we adapt the configuration model to the case of tripartite hypergraphs. Apart from that we use it extensively to determine a large number of results in the dissertation.

In addition to the models described here there are a number of other random graph models, most notably Markov graphs and their general forms, exponential random graps [75, 142] as well as the small-world model of Watts and Strogatz [149].

## 1.6 Network growth models

In the previous section we described examples of models, where we take observed properties from real networks, such as the degree sequence or clustering, and then use that as an input-parameter to the model to generate networks that share similar properties. These models however, do not tell us much about why networks have these properties to begin with. Consequently, there are a different class of models—collectively referred to as network evolution or growth models—that try to explain the origin of these properties on the basis of some kind of evolution process. The typical approach is to let the network grow by the addition of vertices and edges in some fashion, intended to reflect the growth process that might actually be taking place on real networks. The hypothesis is that the growth process itself is what gives a network its characteristic structural features.

Among these growth models, by far the most widely studied and influential are those that aim to explain the origin of the highly right-skewed degree distributions that are ubiquitous in real networks. In fact, one can argue that the recent widespread interest in networks has been inspired in a major way by these models and its implications.

In this section we will focus on the two most famous examples; the archetypal model of Price [48, 49] and the closely related model of Barabási and Albert [13] that has been the inspiration for much of the work in this area.

### 1.6.1 The model of Price and Barabási-Albert

The first example of a right-skewed, or what is now known as a scale-free network, was provided in 1965 by the English physicist and later science historian Derek de Solla Price. Price studied the network of citations between scientific papers, and found that distributions for both the number of times a paper is cited, as well as the number of other papers that a paper cites (the in-degree and out-degree) appear to follow power-laws [48]. Some years later he published another paper in which he proposed an explanation for the origin of these power-laws [49]. His work was based on the seminal work done by the noted economist, sociologist and general polymath Herbert Simon who in the 1950's showed that power-laws arise by the so-called "rich-get-richer" mechanism, where, the amount one gets, goes up as the amount one already has. Simon was referring to wealth distributions, and later on it was applied to other concepts, for example in sociology it is amusingly referred to as the *Matthew effect*[4]. Price however, was the first to apply this concept to a networked system, and in that context he called it *cumulative advantage*.

His basic idea was that the rate at which a paper receives citations should be proportional to the number of times it already has been cited. This is not too hard a proposition to swallow. It seems reasonable that the probability that we come across a particular paper, while reading the literature should increase with the number of citations to it (in fact that's probably how we came across it in the first place) and similarly the probability that we cite the paper in a paper we write should increase due to the same reasons. A similar argument can be applied to the Web, in the context of web-pages that link to each other (one can think of a hyperlink as a citation). Although we can assign complex forms for the dependency of the citation probability on previous citations, the simplest choice is that it is linear, and this is what Price assumed. The following is the description of the model.

Consider a directed graph of $n$ vertices, and let $p_k$ be the fraction of vertices with in-degree $k$. The network evolves by the addition of new vertices each with a fixed out-degree—the number of papers it cites. The out-degree is allowed to vary from vertex to vertex and can take on non-integer values, however the mean value, let's call it $\mu$, is fixed. Clearly $\mu$ is also the mean in-degree for the network, i.e. $\mu = \sum_k kp_k$.

The probability of a newly arriving edge to attach to a previously extant vertex—

[4]From the biblical verse: *For to every one that hath shall be given* . . . Matthew(25:26)

the act of citing a paper—is assumed to be proportional to the in-degree of the vertex. There is, however, a flaw in the formulation. As the model stands, each new vertex has in-degree zero and therefore the probability of it receiving any new edges will forever remain zero. Price proposed a reasonable fix for this. He suggested that the probability of attachment be proportional to $k + k_0$ where $k_0$ is some fixed value that corresponds to the initial number of citations that a paper enters the network with. A fair suggestion is that $k_0$ be one, which can be interpreted as the initial publication of a paper is its first citation (to itself). We can thus write down the probability of a freshly arrived edge to attach to a vertex of degree $k$, that is already part of the network as,

$$\frac{(k+1)p_k}{\sum_k (k+1)p_k} = \frac{(k+1)p_k}{\mu+1}, \tag{1.53}$$

where the denominator is included to make sure that it is properly normalized.

At any given time the the number of vertices with in-degree $k$ is given by $np_k$. The mean number of edges added to each vertex is $\mu$ and thus the mean number added to vertices with in-degree $k$ is just $\mu$ times the attachment probability of Eqn. (1.53). This implies that vertices with degree $k$ now have degree $k+1$ and thus $np_k$ decreases by this amount. On the other hand there is also a corresponding increase, due to the influx from vertices that had previously $k-1$ edges and now have $k$. Vertices with degree zero have to be treated separately as they have an influx of exactly one.

If for a network of $n$ nodes, we denote the fraction of nodes that have degree $k$ as $p_{k,n}$, then the change in that value after the addition of new vertices and edges can be written with the help of a rate equation (or master equation) thus,

$$(n+1)p_{k,n+1} - np_{k,n} = \begin{cases} [kp_{k-1,n} - (k+1)p_{k,n}] \times \mu/\mu+1 & \text{for } k \geq 1, \\ 1 - p_{0,n} \times \mu/\mu+1 & \text{for } k = 0. \end{cases} \tag{1.54}$$

Assuming the distribution tends to a stationary state, we set $p_{k,n+1} = p_{k,n} = p_k$, and then Eqn. (1.54) can be re-arranged to give,

$$p_k = \frac{k(k-1)\dots 1}{(k+2+1/\mu)\dots(3+1/\mu)} \times p_0 = (1+1/\mu)B(k+1, 2+1/\mu), \tag{1.55}$$

where $B(a,b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$ is the Euler Beta function (also called the Legendre function by some authors) [1]. It can be shown that for fixed $b$ and large $a$, $B(a,b)$ goes asymptotically as $a^{-b}$, which implies that for large values of $k$—that is in the tail of the distribution,

$$p_k \sim k^{-(2+1/\mu)}. \tag{1.56}$$

In the limit of large $n$ then the tail of the distribution behaves as a power-law $p_k \sim k^{-\alpha}$ with an exponent $\alpha = 2 + 1/\mu$. This typically leads to exponents between the range $2 \leq \alpha \leq 3$, which is similar to what is seen in the real world. Price himself compared his results to data taken from the Science Citation Index, and found good agreement. Note, that it is fairly straightforward to extend the results to the case $k_0 \neq 1$, however the form of the exponent is the same.

Although Price's model of cumulative advantage is now widely accepted as the most probable explanation for why networks such as citation nets or the World Wide Web have power law degree distributions, his results were largely unknown in the scientific community, until its rediscovery some twenty odd years later by Barabási and Albert who applied the same idea to a different setting from what Price did. During the 1990s it was widely surmised that pages on the World Wide Web, owing to its decentralized nature, are connected purely at random, meaning that the degree distribution of web-pages should resemble a Poisson distribution. Empirical measurements showed [60] that this was far from the truth, and in fact the distributions were highly right-skewed and resembled power-laws. To explain this phenomenon, in a highly influential paper [13], Barabási and Albert proposed a mechanism virtually identical to Price's—with some subtle differences as we will see shortly—to which they gave the name *preferential attachment* (we will refer to this as the BA model for short).

In the BA model, as in Price's, vertices are added to the network with a fixed value $\mu$, which is never changed thereafter. However the difference between the two models is that in the BA model, edges are assumed undirected. This has the benefit that it gets round the problem that Price faced—regarding how a paper gets its first citation—since each vertex in the graph arrives with initial degree $\mu$ and thus has a non-zero probability of receiving new links. Although from a technical standpoint, in order for the model to be solvable using the master-equation method, $\mu$ is the same for every vertex and is not allowed to vary around some mean value. Another drawback of the model, is that by assuming that edges are undirected, it sacrifices realism, since the Web really is a directed graph. Nevertheless the model has the virtue of simplicity, and it does approximate well the behavior of power-law networks. Moreover its importance can be attributed more to the subsequent work that it inspired, rather than the accuracy of its results.

The model is exactly solvable in the limit of large size, Barabási and Albert themselves gave an approximate solution, but subsequently, Krapivsky *et al.* [96] and independently Dorogovtsev *et al.* [54] have used the rate equation method to solve for a number of its properties.

The equivalent expression for Eqn. (1.53), the probability that a new edge attaches to an existing vertex is,

$$\frac{kp_k}{\sum_k kp_k} = \frac{kp_k}{2\mu}, \tag{1.57}$$

where the mean degree of the network is now $2\mu$ since each vertex added has $\mu$

edges, and since the edges are now undirected and have two ends they contribute twice to the mean value for the network. Keeping in mind that in this case there are no vertices with degree $k < \mu$ and accounting for for the change in the attachment probability, the corresponding rate equation for the evolution process can be written as,

$$(n+1)p_{k,n+1} - np_{k,n} = \begin{cases} \frac{1}{2}[(k-1)p_{k-1,n} - kp_{k,n}] & \text{for } k > 1, \\ 1 - \frac{1}{2}\mu p_{\mu,n} & \text{for } k = \mu. \end{cases} \tag{1.58}$$

Then in a similar fashion as in Eqn (1.54), we seek a stationary solution to the equation above, and after some algebra it can be shown that,

$$p_k = \frac{(k-1)(k-2)\ldots\mu}{(k+2)(k+3)\ldots(\mu+3)} \times p_\mu = \frac{2\mu(\mu+1)}{(k+2)(k+1)k}, \tag{1.59}$$

which in the limit of large $k$ goes as,

$$p_k \sim k^{-3}. \tag{1.60}$$

As we can see an additional limitation of the model, when compared to Price's, is that it produces only a single value for the exponent $\alpha = 3$. Nevertheless, the simplicity of the model lends itself well to analytical calculation and a variety of authors have provided solutions for other properties. For example, Krapivsky and Redner [93] have conducted a detailed analytical study of the model and among other things have demonstrated that it has two important correlations. First, there is a correlation between the age of vertices—the time that a vertex is present in the network measured on some time-scale—and their degrees. They have also demonstrated that vertices have degree correlations of the type discussed in Sec. 1.4.4. There are also a number of papers describing the results of extensive numerical simulations of the model, notable among them are those by Dorogovtsev and Mendes [50] and Krapivsky and Redner [94].

### 1.6.2 Extensions to the Barabási-Albert model

As we mentioned before, the BA model has received an exceptional amount of attention since its publication, and over the last few years a number of authors have extended the model to better reflect the behavior of real networks. An extensive review of these developments is provided in [6], here we describe some notable examples.

We saw in the previous section that the BA model is limited by the fact that it only produces a single value of the exponent, $\alpha = 3$. To remedy this, Dorogovtsev and collaborators [54] as well as Krapivsky and Redner [93] (most readers will by now recognize that these two authors have been extraordinarily prolific in dealing with this subject matter), have proposed and studied an extension to the model, where the attachment probability is proportional to $k + k_0$, where $k_0$ is a fixed offset as in Price's model. Unlike the Price model, however, $k_0$ is allowed to be negative and can take on any value between the range $-\mu < k_0 < \infty$. Through a similar sequence of arguments leading upto Eqns. (1.55) and (1.59), the solution for the degree distribution $p_k$ is,

$$p_k = \frac{(k-1)\ldots\mu}{(k+2+k_0/\mu)\ldots(\mu+3+k_0/\mu)} \times p_\mu = \frac{B(k, 3+k_0/\mu)}{B(m, 2+k_0/\mu)}, \tag{1.61}$$

where as before $B(a,b)$ is the Beta function. in the large $k$ limit this gives a power-law $p_k = k^{-\alpha}$, with an exponent of the form $\alpha = 3 + k_0/\mu$. If the value of $\mu$ is negative, then this leads to values for $\alpha < 3$ seen in many real world networks.

In all the variations of the model described so far, the probability of attaching to a vertex has been assumed to be linear in its degree $k$. Krapvisky *et al.* [96, 93] have extended the model to the case where the attachment probability goes as some general power of the degree $k^\gamma$. They found that the model displays three general classes of behavior. For $\gamma = 1$ the results are the same as for normal linear preferential attachment. For $\gamma < 1$, the degree distribution is a power-law multiplied by a stretched exponential, whose exponent is a complex function of $\gamma$. For $\gamma > 1$, there is a condensation phenomenon, in the sense that a single vertex has an extensive connection to all vertices in the network, while for $\gamma > 2$, there is a non-zero probability for a single vertex to connect to every other node in the network.

Dorogovtsev and Mendes have considered the case where the mean degree of vertices added are allowed to vary in time [51]. In particular they assume that the number of new edges added $\mu$ is allowed to grow as a function of the the number of nodes in the networks $n$ as $n^a$ for some constant $a$, and the attachment probability is proportional to $k + Bn^a$ for constant $B$ (not to be confused with the beta function $B$ discussed before). They conclude that the degree distribution is a power-law with exponent $\alpha = 2 + B(1 + a)/(1 - Ba)$. Note, that the constants $a, B$ can be tuned to match the observed values of the exponents in real world networks.

A crucial feature missing in all these models, is that vertices and edges are only added to the network and there is no accounting for the deletion of these. In the real world, such as in the case of the Web, it is a matter of common experience, that web-pages are both added as well as deleted. The same applies to a variety of other networks. To account for this, Sarshar and Roychowdhury [136] and independently Chung and Lu [40] and Cooper *et al.* [43] studied a model of preferential attachment, where in addition to the deposition of vertices and edges, the network was also allowed to lose these in some random fashion. While they provided approximate results for the asymptotic limit in the tail, Moore *et al.* [103] provided exact solutions for the degree distribution for a variety of addition schemes, in addition to preferential attachment—the details of which is the subject of Chapter IV.

There are a number of other interesting extensions to the model, see for example [2, 5, 22, 143, 98].

## 1.7 Percolation and network resilience

A subject that has received much attention in the network literature is the study of percolation processes. In the context of networks, a percolation process is one in which vertices or edges are deemed "occupied" or "unoccupied" in some fashion—random or otherwise—and then one examines the properties of the resulting patterns of connections. If we focus only on the vertices, then the process is referred to as *site percolation*, instead if we consider the edges, then it is referred to as *bond percolation*, and as one can imagine, we can consider a combination of both. Percolation theory will be familiar to physicists working in condensed matter theory, where it has been the subject of intense scrutiny [90, 46, 140]. One of the main motivations for its formulation, however, was to model the spread of diseases [28, 73, 70], and in fact it was first applied to networks in a similar context [118]. Most work on percolation, however, has focussed on the subject of network resilience. Consider the following as motivation.

As discussed in Sec. 1.4.6, most networks consist of a set of connected components, with the largest set being the giant component. If we consider a simple communication network for example, in which the existence of a path between any two vertices means that the two can communicate with each other, then those vertices in the giant component can communicate with an extensive fraction of the network, while those in the small components have a limited set of fellow nodes that they can reach. A particularly simple way of measuring resilience then, is to progressively eliminate vertices (or edges), and then examine the variation (or not) in the fraction of vertices that are part of the giant component. In this context, a variety of numerical studies have been conducted on real networks [30, 8], the results of which led to the realization that the problem of resilience of networks to failures—node or edge removal—is equivalent to studying site/bond percolation on the network.

Using networks generated by the configuration model as a template on which to simulate this process, a number of analytical results are known [41, 34]. While [41] considered the case of random site percolation, here we present some details from the more general formulation of [34] who using generating function methods extend the analysis to both site and bond percolation as well as the case of non-random deletion.

Let $p_k$ represent the fraction of vertices in a network that have degree $k$, and $r_k$ be the probability that a vertex is occupied—that it hasn't been removed from the network—given that it has degree $k$. Note, that $r_k$ can take on a variety of forms. For example, if we choose to remove vertices with degree above a certain threshold value $k_{max}$, then $r_k = \theta(k - k_{max})$, where $\theta(x)$ is the Heaviside step function; random removal implies, $r_k = \phi$, where $\phi$ is some constant. The total probability that a vertex of degree $k$ is occupied, is then $r_k p_k$, and we define a generating function for this distribution thus,

$$F_0(z) = \sum_{k=0}^{\infty} r_k p_k z^k, \tag{1.62}$$

where $F_0(1) = r$ is the total fraction of vertices that are occupied. Similarly we can write down a generating function for vertices that are reached by following a randomly chosen edge,

$$F_1(z) = \sum_{k=0}^{\infty} r_{k+1} q_k z^k, \tag{1.63}$$

where $k$ now is the excess-degree and $q_k$ is the excess degree distribution as discussed in Sec. 1.5.2. Using a similar sequence of arguments leading upto the development of Eqns. (1.44) and (1.45), it can be shown that the size distribution of components is generated by $H_0(z)$, where,

$$H_0(z) = 1 - F_0(1) + zF_0(H_1(z)), \quad H_1(z) = 1 - F_1(1) + zF_1(H_1(z)). \tag{1.64}$$

The term $1 - F_0(1)$ represents the case where we pick a vertex at random that is not occupied to begin with (and thus not part of any component), while $1 - F_1(1)$ represents the same for the case of picking a random edge and following it to one of the vertices at its ends. Using $H_0$ and $H_1$ we can calculate a variety of properties, for example the mean size of a component to which a randomly chosen vertex belongs to is given by,

$$\langle s \rangle = F_0(1) + \frac{F_0'(1)F_1(1)}{1 - F_1'(1)}, \tag{1.65}$$

from which we can see that the point marking the emergence of the giant component is given by $F_1'(1) = 1$. In much the same way that we derived the equations for the size of the giant component for the "static" configuration model, it can be shown that the corresponding equations in this case are,

$$S = F_0(1) - F_0(u), \tag{1.66}$$

with $u$ the solution to,

$$u = 1 - F_1(1) + F_1(u). \tag{1.67}$$

Note that in the case of random deletion, $r_k = \phi$, $F_0(z) = \phi G_0(z)$ and $F_1 = \phi G_1(z)$, where $G_0(z), G_1(z)$ are the standard generating functions (1.42).

More detailed versions of the calculations shown here, as well as those for other properties can be found at [34]. In Chapter II we use similar methods to extend the calculation to more complicated definitions of resilience.

## 1.8 Outline of the dissertation

We have described so far a number of structural measures, analytical tools as well as mathematical models that are typically employed in analyzing the properties of networks. In this dissertation, with the aid of these, we examine a diverse set of new problems related to both the structural and dynamical properties of a variety of model as well as real networks. We formulate our theory using a combination of detailed analytical and computational techniques, and test them rigorously on artificially generated model networks. Finally we compare the predictions and results of our theory against data gathered from real networks, and find that in most cases, the theory is a good approximation to the real world.

The material dealt with in the dissertation is roughly divided into three parts. Chapters II and III deal with structural aspect of networks, Chapters IV and V focus on network dynamics and finally Chapter VI seeks to combine a bit of both. While the first two parts employ a combination of theory, simulation and empirical measurements, the last part is based on computer simulations in the spirit of what is conventionally referred to as *agent based modelling* [16].

As we discussed in the previous section, the traditional way to measure the resilience of a network, is to examine the size of the giant component as a function of various schemes of node and edge removal. This can be thought of as simple model for the performance of a communication network such as the World Wide Web under the failure or removal of web-pages. In the real world, however, it is considered fallacious to depend on simple connections between vertices, and typically infrastructural and communication networks rely on redundant connections, such that the failure of any one connection between two vertices does not disrupt performance.

This concept of redundancy leads us to define network resilience in terms of more complicated structures such as bicomponents—sets of vertices that are connected by at least two paths such that the failure of any one path does not disconnect the set. In Chapter II we study these bicomponents, and with the aid of analytical techniques and computer simulations calculate a number of properties of interest. In particular we calculate the size of the giant bicomponent and the conditions for its existence, as well as measure its size as a function of vertex and edge removal. We then compare our results to a variety of real world networks to test our predictions.

The material in Chapter III is inspired by the emergence of new types of social networks called folksonomies. These are tripartite structure of users, resources, and tags—labels collaboratively applied by the users to the resources in order to impart meaningful structure on an otherwise undifferentiated database. We propose a mathematical model of such tripartite structures which represents them as random hypergraphs. We show that it is possible to calculate many properties of this model exactly in the limit of large network size and we compare the results against observations of a real folksonomy, that of the on-line photography web site Flickr.

In Chapter IV we change focus, and build upon the ideas discussed in Sec. 1.6.1. We formulate a model of a network that evolves under the addition and deletion of vertices and edges, and solve for the equilibrium degree distribution for a variety of cases of interest, including preferential attachment. Using similar methods, in Chapter V we consider networks whose structure can be manipulated by adjusting the rules by which vertices enter and leave the network. We focus in particular on degree distributions and show that, with some mild constraints, it is possible by a suitable choice of rules to arrange for the network to have any degree distribution we desire. In addition we define a simple local algorithm based on biased random walks by which appropriate rules can be implemented in practice. As an example application of our ideas, we describe and simulate the construction of a candidate network with certain optimized properties of interest.

In Chapter VI, we formulate a model of interacting agents, that seek to optimize their closeness centrality (as discussed in Sec. 1.4.5), while at the same time keep the number of connections (their degree) low. The ideas are inspired by similar models found in game theory, where agents (as the participants of the game are called) progress in the game by balancing a set of competing interests. With the aid of extensive computer simulations we track the performance of the agents as well as a number of structural network properties as a function of time. We propose that our model captures some of the features in the real world, such as a set of political lobbyists or diplomats who compete with each other to rise to a position of influence.

A brief summary of our results described in Chapter VII will conclude this dissertation.

Note, that each chapter is written in a self-contained fashion, such that if a particular chapter catches a reader's fancy, he or she can "get to meat of the matter", as it were, without necessarily having to read any of the other chapters. However, when necessary, we refer the reader to the appropriate material described in this introduction. The work presented here is adapted from a combination of previously published papers and book chapters—see [63, 64, 65, 76, 77, 103, 116].

CHAPTER II

# Network resilience: The case of bicomponents and $k$-components

## 2.1 Introduction

The issue of the robustness of connections in networks has received considerable attention from the physics community, as an elegant application of percolation theory and other concepts of statistical physics, to a problem of substantial practical importance in many areas such as communication networks and epidemiology [8, 41, 34]. The typical approach is to consider the largest set of vertices in a network that are connected to one another by at least one path—the giant component—and examine how its size varies as vertices are removed from the network. This can be thought of as a simple model for the performance of, for instance, a communication network such as the Internet under failure of vertices. The vertices removed can be chosen at random [41], as if failing because of random technical faults, or in a targeted fashion [34], as if an adversary were deliberately removing them in an effort to destroy network connectivity.

In the real world, however, it is often considered inadequate to rely for communication on just a single path between vertices. It is a well known fact that large organizations such as corporate entities maintain at least two or more independent data connections to the internet, such that the failure of any one connection does not leave them disconnected from the network. Two examples with rather different outcomes spring to mind. In August of 2003, there was a major power outage in the Northeastern part of the United States whose effects quickly spread to some of the Midwestern states, as well as parts of Ontario, Canada [18]. Investigations revealed that the fault was caused by a disruption in the power lines, that led to a series of cascading failures resulting in a major blackout. One of the main reasons for the failure was the lack of excess capacity in the power cables as electricity re-routed through alternative lines [89]. An example with a happier outcome is provided by the case of the Fiber-Optic Link around the Globe (FLAG), which is a 28,000-kilometre long submarine communications cable containing optical fiber, that supports much of the world's Internet bandwidth. In early 2008, the failure of cables in the Mediterranean sea, led to disruption of Internet services in the Middle East and the Indian subcontinent, which had the potential to lead to major financial losses, as a significant fraction of Information Technology and financial back office operations for companies in the United States are provided by firms in India [144]. However, the Indian companies had accounted for such a situation, and were able to re-route their services via lines in the Pacific Ocean, thus keeping service disruptions to a minimum, a fact that contributed in no small measure to their reputation.

This concept of redundancy, leads us to an obvious generalization of robustness in networks, in which we focus on the set or sets of vertices in a network that are connected by two or more independent paths (in the sense that they share no common vertices) such that the failure of any one vertex does not disconnect the set(s). Such sets are called *bicomponents*.

In this chapter we study the resilience of networks to disruptions, by studying its bicomponent structure, in both real and model networks. In particular, using standard network models, we show that there exists at most one giant bicomponent in a given network that is nested within the regular giant component, and whose appearance coincides with the emergence of the giant component. We also provide expressions for the expected size of the giant bicomponents and higher $k$-components along with their variation as a function of vertex removal. In addition we compare our theoretical results to data gathered from a variety of networks in the real world.

## 2.2 Bicomponents

### 2.2.1 Basic properties

Consider two paths that connect the same pair of vertices in a network. They are said to be *vertex-independent*, or simply *independent*, if they share none of the same vertices other than the starting and ending vertices. A $k$-component is a maximal subset of the vertices of a network such that every vertex in the set is connected to every other by $k$ independent paths [151]. For the special cases of $k = 2, 3$ the $k$-components are also called *bicomponents* and *tricomponents*, respectively. Note that not all vertices need belong to a $k$-component for $k \geq 2$, which contrasts with the case for ordinary components ($k = 1$), where every vertex belongs to a component.

The vertices in a bicomponent have the property that no two can be disconnected by the failure of any other single vertex. Another observation that will become important shortly is that the $k$-components of a network are nested. That is, every bicomponent is, trivially, a subset of an ordinary component, every tricomponent is a subset of a bicomponent–since there are at least two independent paths in a tricomponent, and so forth—see Fig. 2.1 for an example illustration. In what is to follow we will primarily concentrate on bicomponents, although we will give some results for $k \geq 3$ where appropriate.

To gain an understanding of the expected behavior of the bicomponents in a

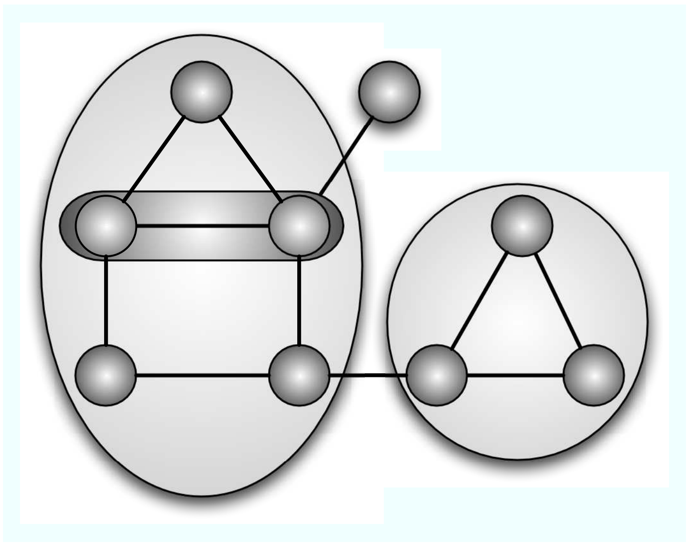

Figure 2.1: A network containing $k$-component structure. The nested property of $k$-components can be seen clearly. In the example above, the entire network is a regular 1-component, within which are nested the bicomponents (lighter shade). In the left side of the figure, we can see a tricomponent (darker shade) nested within a bicomponent. Note the isolated vertex, that does not belong to any component for $k \geq 2$.

network, we will make use of the configuration model for unipartite graphs (cf. Sec. 1.5.2). As discussed before, the probability of an edge falling between two vertices $i$ and $j$ in such a network is given by the expression,

$$P_{ij} = \frac{k_i k_j}{2m}, \tag{2.1}$$

where $k_i$ is the degree of vertex $i$ and $m$ is the total number of edges in the network. For a sparse network—where the network is allowed to grow to large size, while keeping the average degree fixed—the model predicts the existence of at most one giant regular component (1-component) of size $\mathrm{O}(n)$ and many small components of size $\mathrm{O}(1)$, a pattern that is seen in most real-world networks as well [102, 101, 117].

A first interesting result to note is that, by contrast with the case for 1-components, there are in general (almost) no small bicomponents in the configuration model. To see this consider the small 1-components of the configuration model, which, are generally tree-like, meaning they are not bicomponents. In order to turn them into bicomponents, we would need to add at least one edge to the tree, thereby closing a loop and creating two paths between some vertices. Since the small components have size O(1) and hence also O(1) pairs of vertices, there are O(1) opportunities to perform such a closure in each small component. Each closure occurs with probability $P_{ij}$ as in Eqn. (2.1), which is of $\mathrm{O}(n^{-1})$ on a sparse network and hence the total probability of converting a small component into a bicomponent is $\mathrm{O}(n^{-1})$. Since there are $\mathrm{O}(n)$ small components, this means that the total number of small bicomponents formed in this way is O(1). Small bicomponents can also be formed out of tree-like subsets of the giant component, but a similar argument shows that such bicomponents are also O(1) in number.

Thus in the limit of large network size the probability that a randomly chosen node belongs to a small bicomponent vanishes as $1/n$. Speaking loosely, there are no small bicomponents in the network.

### 2.2.2 Size of the giant bicomponent

There may, however, be a *giant bicomponent*, and moreover it turns out to be possible to calculate the expected size of the giant bicomponent exactly in the limit of large graph size. In order for a vertex to belong to the giant bicomponent at least two of the edges incident on that vertex must lead to the giant bicomponent by independent paths. However, since the giant bicomponent is a subset of the giant 1-component, it follows that any edge that leads to the giant 1-component also leads to the giant bicomponent, so an equivalent statement is that at least two of our vertex's neighbors must be in the giant 1-component.

The probability that the neighbor of a vertex belongs to the giant component is straightforward to calculate [117]. Let the degree distribution of our network be $p_k$, meaning that a randomly chosen vertex has degree $k$ with probability $p_k$. If, however, we choose an edge at random and follow it to one of the vertices at its ends, then the number of edges incident on that vertex, other than the one we arrived along, follows the excess degree distribution:

$$q_k = \frac{(k+1)p_{k+1}}{\langle k \rangle}, \tag{2.2}$$

as shown in [117], for example. Here $\langle k \rangle = \sum_k kp_k$ is the mean degree for the entire network.

At this point it will be convenient to define the following generating functions for the degree and excess-degree distributions,

$$G_0(z) = \sum_{k=0}^{\infty} p_k z^k, \quad G_1(z) = \sum_{k=0}^{\infty} q_k z^k. \tag{2.3}$$

Let $u$ be the probability that upon following an edge we arrive at a vertex that does not belong to the giant component, then it must be that none of the other edges emanating from that vertex leads to the giant component. If $k$ denotes the excess-degree of that vertex then, the probability for this to happen is $u^k$. Averaging over the distribution $q_k$, we have,

$$u = \sum_k q_k u^k = G_1(u). \tag{2.4}$$

In order for there to be a giant component there must be a solution to Eq. (2.4), where $u < 1$. It is a fairly easy exercise to show that this condition is satisfied, provided $G_1'(1) > 1$, which leads to the well know criterion of Molloy and Reed for the existence of the giant component [102].

Armed with these results it is now a simple task to write down the probability that a randomly chosen vertex belongs to the giant bicomponent. The probability that it *does not* is the probability that either zero or one, but not more, of its edges lead to vertices in the giant component, which is

$$\sum_k p_k u^k + \sum_k k p_k u^{k-1} = G_0(u) + (1-u)G_0'(u). \tag{2.5}$$

Thus, the probability that the vertex *does* belong to the giant bicomponent is just one minus this quantity,

$$S_2 = 1 - G_0(u) - (1-u)G_0'(u), \tag{2.6}$$

where $S_2$ can also be thought of as the expected size of the giant bicomponent as a fraction of the size of the entire network.

We have not, in this derivation, demonstrated that the two paths from our vertex to the giant bicomponent are independent. However, since the diameter of a random graph is $\mathrm{O}(\ln n)$ [11], and since the diameter provides an upper bound on the lengths of the paths in our derivation, the probability of their intersecting is $\mathrm{O}(n^{-1}\ln^2 n)$, which vanishes as $n \to \infty$, so our result will be correct provided again that our networks are large.

There can of course be no giant bicomponent when there is no giant 1-component, since the former is a subset of the latter. Equation (2.6) shows that, in general, the reverse is also true: when there is a giant 1-component there is also a giant bicomponent. Only in the special case where

$$1 - G_0(u) = (1-u)G_0'(u), \tag{2.7}$$

can the giant bicomponent vanish. Thus the giant bicomponent appears, in general, at the same time as the giant component: for any specific family of degree distributions, if there is a phase transition marking the appearance of the giant component, the same transition point also marks the appearance of the giant bicomponent.

It is a straightforward exercise to generalize the expression for the size of the giant bicomponent to any $k$-component. For example if we consider the case of tricomponents, there will be an additional term in Eq. (2.5), which includes the case where two edges lead to the giant component, but excludes higher ones. Similarly for the four-component, there will be a fourth term for three edges leading to the giant component. It is then easy to see that the expression for the size of the $k$'th component is given by,

$$S_k = 1 - \sum_{m=0}^{k-1} \frac{(1-u)^m}{m!} \frac{d^m G_0}{dz^m}\bigg|_{z=u}, \tag{2.8}$$

which implies that in fact the transition marking the appearance of all $k$-components coincide with that of the ordinary giant component.

### 2.2.3 Percolation

As the presence of $k$-components in a network is a desirable property, it is instructive to ask how robust this property is to the removal of vertices or edges. In the previous section we established the size of the giant bicomponent and other higher $k$-components, and found that they all occur at the same time as the giant component, in other words as we start removing vertices the threshold for the destruction of connectivity for all orders coincide. However, this does not necessarily tell the whole story. Using a simple variant of the arguments above we can also calculate the size of the giant $k$-component as vertices are removed.

Let $r_k$ be the probability that a vertex of degree $k$ is operational (i.e., it hasn't failed or been removed from the network). Then the probability that the vertex has degree $k$ and is occupied is just $p_k r_k$. Common choices for $r_k$ are $r_k =$ constant, which corresponds to uniformly random failure of vertices, or $r_k = \theta(k_{\max} - k)$, where $\theta(x)$ is the Heaviside step function, which corresponds to removal of all vertices with degree $k > k_{\max}$, a form of targeted attack against the best-connected vertices [8].

The corresponding probability generating functions are then,

$$F_0(z) = \sum_{k=0}^{\infty} p_k r_k z^k, \quad F_1(z) = \sum_{k=0}^{\infty} q_k r_{k+1} z^k. \tag{2.9}$$

It is then straightforward to see that the probability $u$ that an edge does not lead to such a vertex in the giant component, is a sum of two terms, the first corresponding to the vertex being not operational, or $1 - r_{k+1}$, and the second representing the fact that it is operational but none of its remaining edges lead to the giant component, i.e , $r_{k+1}u^k$. Averaging over the distribution gives us,

$$u = 1 - F_1(1) + F_1(u). \tag{2.10}$$

The calculation of the expected size of the giant $k$-component is then the same as in equations (2.5) and (2.6), with the generating function $G_0(u)$ replaced by $F_0(u)$ taking into account the vertex occupation probability. The general expression for the giant $k$-component is then simply,

$$S_k = F_0(1) - \sum_{m=0}^{k-1} \frac{(1-u)^m}{m!} \frac{d^m F_0}{dz^m}\bigg|_{z=u}. \tag{2.11}$$

The percolation transition at which the giant 1-component in a network is destroyed is typically second-order in nature. For $k$-components with $k \geq 2$ on the other hand, we can show using the results above that the transition is of higher order. Consider for instance the case of uniform random failure of vertices, for which $r_k = \phi$ independent of $k$. Then Eq. (2.10) simplifies to,

$$u = 1 - \phi + \phi G_1(u). \tag{2.12}$$

Performing a Taylor expansion of $G_1(u)$ around the point $u = 1$, writing $u = 1 - \epsilon$ and making use of the fact that $G_1(1) = 1$, we can write $\epsilon$ as,

$$\epsilon = \phi[\epsilon G_1'(1) - \tfrac{1}{2}\epsilon^2 G_1''(1)] + \mathrm{O}(\epsilon^3). \tag{2.13}$$

This can be rearranged to give us,

$$\epsilon = \frac{2[\phi G_1'(1) - 1]}{\phi G_1''(1)}, \tag{2.14}$$

which tells us that $1-u$ is linear in the term $\phi - \phi_c$, where $\phi_c = 1/G_1'(1)$ marks the point at which the giant component emerges [41].

Referring back to Eq. (2.11), we can recast it in the following manner. Performing a Taylor expansion of the function $F_0(z)$ around the point $z = u$, leads to,

$$F_0(z) = \sum_{m=0}^{\infty} \frac{(z-u)^m}{m!} \frac{d^m F_0}{dz^m}\bigg|_{z=u}. \tag{2.15}$$

Setting $z = 1$ and making use of (2.11), we get a difference of two sums which can be re-written as,

$$S_k = \sum_{m=k}^{\infty} \frac{(1-u)^m}{m!} \frac{d^m F_0}{dz^m}\bigg|_{z=u}. \tag{2.16}$$

The leading order term in Eq. (2.16) for a given $k$ is of order $(1-u)^k$. Combining this with our result from Eq. (2.14) we see that the expected size of a $k$-component $S_k$ is of $\mathrm{O}(\phi - \phi_c)^k$.

The equation above has interesting implications. It suggests that the network undergoes a infinite series of continuous phase transitions marking the appearance of giant components for all $k$, and occurring at the same critical point $\phi_c$. However the phase transitions are not of the same order, the transition for each $k$ being of order $k+1$. What this means is that although, say the giant bicomponent, appears at the same time as the giant component, its initial growth in size is much slower than that of the giant component. Consequently the network does not achieve a high level of robustness, in this sense, until a significant fraction of the vertices are occupied/operational.

Before comparing our theoretical results with data from real networks, we first test it on computer-generated random graphs. Figure 2.2 shows the variation in the size of

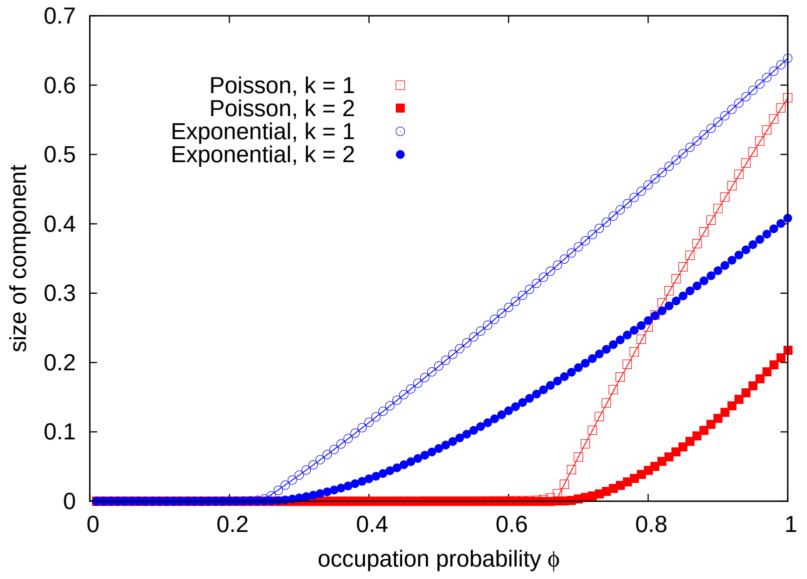

Figure 2.2: Size of the giant component and bicomponent as a function of vertex removal for random networks with exponential($e^{-\lambda k}$ with $\lambda = 0.4$, blue circles) and poisson (mean = 1.5, red squares) degree distributions. Solid lines are the analytical solutions and the points are numerical results for networks of size $10^6$ vertices averaged over a 100 instances of the network. Error bars in all cases are smaller than the size of the points.

the giant (bi)-component as a function of vertex removal for two example networks with poisson and exponentially distributed links. Both networks were generated artificially using the configuration model. The third order phase transition of the bicomponents for both networks can be clearly seen. In addition, the figure shows that the two types of components are destroyed at the same value of $\phi$.

## 2.3 Comparison with real world data

Let us now examine the bicomponent structure in real-world networks. The standard and optimal method for finding bicomponents in a network is the the depth-first search based algorithm of Hopcroft and Tarjan [80] that runs in time $O(n)$. Table 2.1 summarizes the results of applying this algorithm to a variety of previously documented networks. The table reveals some interesting features. First we note that, with two exceptions, the networks all have large giant bicomponents. Certainly the giant bicomponents are smaller than the giant components, but in each case the networks have a substantial fraction of robust connections in the sense considered here. The two exceptions are the collaborations of network scientists and the high-school dating network. The former has quite a small giant component, so the giant bicomponent cannot be very large, although it is still quite a small fraction of the giant component size. The dating network, however, is clearly anomalous, having a very substantial giant component but a very small giant bicomponent. It is interesting to consider whether there might be sociological reasons for this anomaly.

Second, we note that in all but two cases the networks have no small bicomponents, or very nearly none. This observation agrees well with our calculations for random graphs above, but is otherwise somewhat surprising. It has recently been observed that most real-world networks contain a high density of short loops [149], a feature that random graphs lack and one that should automatically give rise to bicomponents. Our observations indicate however, that in most cases such bicomponents are attached to the giant bicomponent, rather than forming independent small bicomponents, and that most portions of the network not in the giant bicomponent are tree-like. Apparently the structure of these networks is, in this respect at least, relatively close to that of the corresponding random graphs. This finding stands in sharp contrast with studies of other network properties such as clustering and assortative mixing, in which random graph models have been found to be in poor agreement with those of real networks.

It is also interesting to examine the robustness of the bicomponent structure to removal of vertices—either random or targeted—as we did for our model networks. The Hopcroft–Tarjan algorithm is a poor choice for this calculation, since we would have to perform $n$ runs of the algorithm to find the bicomponents after the removal

| network | $n$ | $S_1$ | $S_2$ | small bicomp. |
|---|---|---|---|---|
| Internet (AS) | 22 963 | 1 | 0.651 | 0.012 |
| world wide web | 325 729 | 1 | 0.414 | 0.076 |
| power grid | 4941 | 1 | 0.615 | 0.062 |
| *C. Elegans* neural | 297 | 1 | 0.949 | 0 |
| *C. Elegans* metabolic | 453 | 1 | 0.934 | 0.049 |
| physics collaborations | 16 726 | 0.829 | 0.588 | 0.243 |
| network scientists | 1589 | 0.239 | 0.084 | 0.634 |
| friendship | 795 | 0.979 | 0.940 | 0 |
| dating | 573 | 0.503 | 0.072 | 0.014 |

Table 2.1: Statistics of a number of real-world networks. The second to fifth columns give the number of vertices in the network, the fractions occupied by the largest component and bicomponent, and the fraction occupied by small components. The networks are, in order, a snapshot of the Internet topology at the autonomous system (AS) level, the symmetrized web graph of a university web site [7], the Western United States power grid [149], the neural [149] and metabolic [55] networks of the nematode *C. Elegans*, coauthorship networks of physicists [106] and network scientists [110], and friendship and dating networks from a study of US school students [19].

of each vertex, for a total running time $O(n^2)$. For the larger networks studied this is prohibitive, so instead we employ a different algorithm, similar in spirit to the fast percolation algorithm of Newman and Ziff [119], that makes use of established methods based on data trees to keep track of bi-connected subgraphs dynamically [152]. Briefly, it works in the following manner.

The algorithm starts with an empty network and adds vertices, rather than taking them away, and avoids finding the bicomponents anew after each addition by calculating instead only the change in the bicomponent structure from the previous step, which is usually minor. The algorithm stores the structure of the components and bicomponents in two separate “forest” data structures (i.e., sets of trees) with one tree for each (bi)component. Vertices within components contain pointers that point to others in the same component, such that by following a sequence of such pointers we can reach the root of the tree, thereby identifying the component uniquely. As each new vertex is added to the network we check in this way to which components its neighbors belong, amalgamating those components if necessary by adding a pointer from the root of one tree to the root of the other. If the added vertex joins neighbors

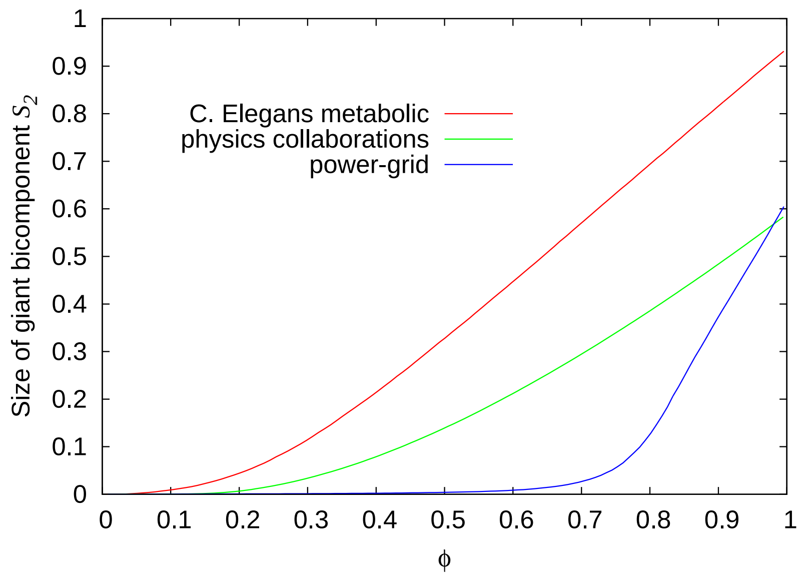

Figure 2.3: Size of giant bicomponent as vertices are randomly removed from three real-world networks, a metabolic network for *C. Elegans* [55], a collaboration network of scientists working in condensed matter phyiscs [106], and the Western States Power Grid of the United States [149].

that already belong to the same component, a loop—and hence new bicomponent—has been created, or old ones extended or joined, and the bicomponent trees are updated appropriately. The tree traversals employed by the algorithm take $O(\ln n)$ time on average and hence the algorithm can add all $n$ vertices in a total running time $O(n \ln n)$. In practice, this gives about a factor 100 improvement in running time or more over the Hopcroft–Tarjan algorithm for the networks studied here, and renders the calculations easily doable on a standard desktop computer.

Figure. 2.3 shows the results of the application of this algorithm to three of the networks from Table 2.1, the metabolic network, the physicist collaborations, and the power grid. For each network there appears to be a transition point below which the giant bicomponent is destroyed and the network can no longer be said to be robustly connected. For two of the networks, however, the transition appears to be at or close to $\phi = 0$, indicating that the networks are highly robust in the sense considered here: nearly all the vertices have to be removed from the network before the giant bicomponent is destroyed. The third network, the power grid, shows a much higher transition probability, indicating that this network is relatively fragile to random node removal. On the other hand, we also see in all cases that the transition at which the giant bicomponent appears is a gradual one; the gradient at the transition is shallow—perhaps even zero—so that the giant bicomponent grows very slowly at first above the transition. In this respect these networks again appear to behave in a manner similar to our random graph models.

## 2.4 Discussion

In this chapter we have proposed an alternative and perhaps more realistic measure of network resilience—its bicomponent and $k$-component structure. Using a combination of analytical and numerical techniques, we have studied bicomponents in both model and real networks. In particular we have shown that for a given network, there are no small bicomponents, however there exists at most one giant bicomponent that is nested within the regular giant component. In addition we have also provided expressions for the expected size of the giant bicomponents and higher $k$-components along with their variation as a function of vertex removal. Finally we have tested our theoretical results on a series of data taken from networks in the real world.

The analytic and numerical results presented give an interesting picture of the behavior of network bicomponents. The real-world networks investigated are found to be quite robust in the sense of having large giant bicomponents and moreover the existence of these bicomponents is, at least in some cases, itself robust to the deletion of vertices. In practice, however, although the giant bicomponent may persist as vertices are removed from the network, its size dwindles rapidly so that large portions of the network lose robust connection considerably before the transition point at which the giant bicomponent finally vanishes. In each of these respects the behavior of our networks is surprisingly similar to the behavior of exactly solvable random graph models, which predict a giant bicomponent that persists down to the point at which the ordinary giant component disappears, but with an unusual third-order transition at that point that ensures that the size of the bicomponent will be small as we approach the transition.

# CHAPTER III

# Random Hypergraphs and their applications

## 3.1 Introduction

In its simplest form a network consists of a set of nodes or vertices, connected by lines or edges, but as discussed in Chapter I, many extensions and generalizations have also been studied, including networks with directed edges, networks with labeled or weighted edges or vertices, and bipartite networks, which have two types of vertices and edges running only between unlike types [108, 24, 53, 113, 32]. In the previous chapter, we saw an example of an unipartite, network. In this chapter, we will look at a more involved representation of a network, a *hypergraph*. As motivation for why we study such systems, consider the following.

In recent times, particularly in the last two or three years, new and more complex types of network data have become available, especially associated with on-line social and professional communities, that cannot adequately be described by existing network formats. One example is the *folksonomy*. Folksonomy is the name given to the common on-line (and sometimes off-line) process by which a group of individuals collaboratively annotate a data set to create semantic structure [35]. Typically mark-up is performed by labeling pieces of data with a short text description called *tags*. A good example is provided by the on-line photography resource Flickr, a web site to which users upload photographs that can then be viewed by other users. Flickr allows any user to give a short description of any photo they see, usually just a single word or a few words. These are the tags. In principle, tags can allow users to do many things, such as searching for photos with particular subjects or clustering photos into topical groups. There are also many other websites and on-line resources with similar tagging capabilities, but dealing with different resources. On the website CiteUlike, for example, users upload academic papers as opposed to photographs and label them with descriptive tags.

Researchers have taken a variety of approaches to the representation of folksonomy data using network methods, including modeling them as simple unipartite graphs and bipartite graphs as well as limited forms of tripartite graphs [69, 36, 97, 122]. Each of these approaches, however, fails to capture some elements of the structure of the data and hence limits the conclusions that can be drawn from subsequent network analysis.

The fundamental building block in a folksonomy is a triple consisting of a *resource*, such as a photograph, a *tag*, usually a short text phrase, and a *user*, who applies the tag to the resource. Any full network representation of folksonomy data needs to capture this three-way relationship between resource, tag, and user, and this leads us to the consideration of hypergraphs.

A *hypergraph* is a generalization of an ordinary graph in which an edge (or *hyperedge*) can connect more than two vertices together. To represent our folksonomy we make use of a *tripartite hypergraph*, a generalization of the more familiar bipartite graph, in which there are three types of vertices representing resources, tags, and users, and three-way hyperedges joining them in such a way that each hyperedge links together exactly one resource, one tag, and one user. Each hyperedge corre-

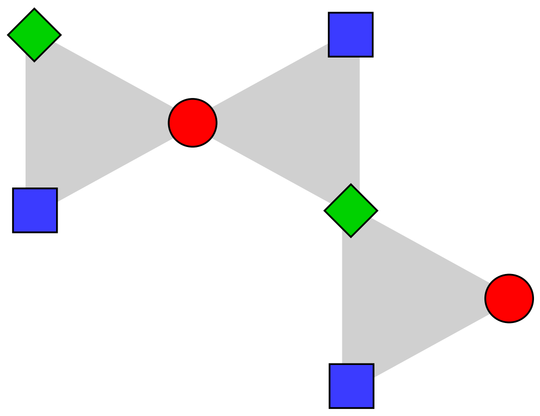

Figure 3.1: Vertices in our networks come in three types, represented here by the red circles, green diamonds, and blue squares, and are connected by three-way hyperedges that each join together exactly one circle, one diamond, and one square. In the language of folksonomies, the circles represent, say, the resources, the diamonds the tags, and the squares the users.

sponds to the act of a user applying a tag to a resource and hence the tripartite hypergraph preserves the full structure of the folksonomy—see Fig. 3.1.

In this chapter, we study the theory of such tripartite graphs, starting with basic network properties such as degree distributions and then developing a random graph model that allows us to make analytic predictions of a variety of network properties. We test our predictions by comparing them with data from the Flickr folksonomy and find good agreement in some, but not all, cases.

## 3.2 Tripartite graphs

We begin our study of tripartite hypergraphs by outlining some of the basic properties of such networks. Our tripartite graphs have three different types of vertices, which, to preserve generality, we will refer to as red, green, and blue vertices. (In this paper, when discussing applications of the theory to folksonomies, red will represent resources, green tags, and blue users, but the theory itself is entirely agnostic about what the colors represent.) Let us suppose that there are $n_r$ red vertices, $n_g$ green ones, and $n_b$ blue ones.

The edges in our network are three-way hyperedges that each connect one red, one green, and one blue vertex. (We might say that the hyperedges are "colorless" or "white," since red, green, and blue make white when combined in the human visual system.) Let us suppose there to be $m$ hyperedges in total.

There are a number of ways in which vertex degree can be defined for a hypergraph. Some authors, for instance, have defined degree as the total number of other vertices to which a given vertex is connected by hyperedges. This corresponds to the definition of degree in an ordinary graph (at least when there are no multiedges or self-edges), but in failing to distinguish between the different types of vertices to which hyperedges are connected, it can lead to confusion in the hypergraph case. The best, and also simplest, definition of degree for a vertex in a hypergraph is simply the number of hyperedges attached to that vertex. Thus a red vertex participating in four hyperedges has degree four. This might mean that it has four green and four blue neighbors in the network, but it is also possible that some neighboring vertices are common to more than one hyperedge, in which case the number of neighboring vertices of a given color may be smaller than four.

The mean degree $c_r$ of a red vertex in our network is given by the number of hyperedges in the network divided by the number of red vertices, and similarly for green and blue:

$$c_r = \frac{m}{n_r}, \qquad c_g = \frac{m}{n_g}, \qquad c_b = \frac{m}{n_b}. \tag{3.1}$$

Rearranging these equations to give three separate expressions for $m$, we also have,

$$n_r c_r = n_g c_g = n_b c_b = m. \tag{3.2}$$

Thus the mean degrees of the different vertex types cannot be chosen independently, but are linked via the fact that the same hyperedges connect to the red, green and blue vertices.

One of the most important parameters of a network is its degree distribution. Just as bipartite networks have two distinct degree distributions, our tripartite ones have three: we define $p_r(k)$ to be the fraction of red vertices in the network that have degree $k$, and $p_g(k)$ and $p_b(k)$ to be the corresponding quantities for green and blue vertices. These distributions satisfy the standard sum rules

$$\sum_{k=0}^{\infty} p_r(k) = \sum_{k=0}^{\infty} p_g(k) = \sum_{k=0}^{\infty} p_b(k) = 1, \tag{3.3}$$

and

$$\sum_{k=0}^{\infty} kp_r(k) = c_r, \quad \sum_{k=0}^{\infty} kp_g(k) = c_g, \quad \sum_{k=0}^{\infty} kp_b(k) = c_b. \tag{3.4}$$

As with bipartite graphs, it is sometimes convenient to form “projections” of tripartite graphs onto a subset of their vertices. In a bipartite graph of red and green vertices, for instance, one forms a projection onto the red vertices alone by constructing the network of red vertices in which vertices are connected by an edge if they share a common green neighbor in the original bipartite graph [117].

While for bipartite graphs there is essentially only one way of performing projections, there are several distinct possibilities for tripartite graphs—see Fig. 3.2. One can again join two red vertices if they share a green neighbor—in our Flickr example from the introduction, two photos would be connected if they have a tag in common. Or one can join two red vertices that share a common blue neighbor—two photos that were tagged by the same user. Or one could join vertices that share *either* a green or a blue neighbor. And of course one can define the equivalent projections onto the green and blue vertices.

But it doesn’t stop there. In a tripartite network, one can also form projections onto two of the colors. For instance, one can form a projected bipartite network

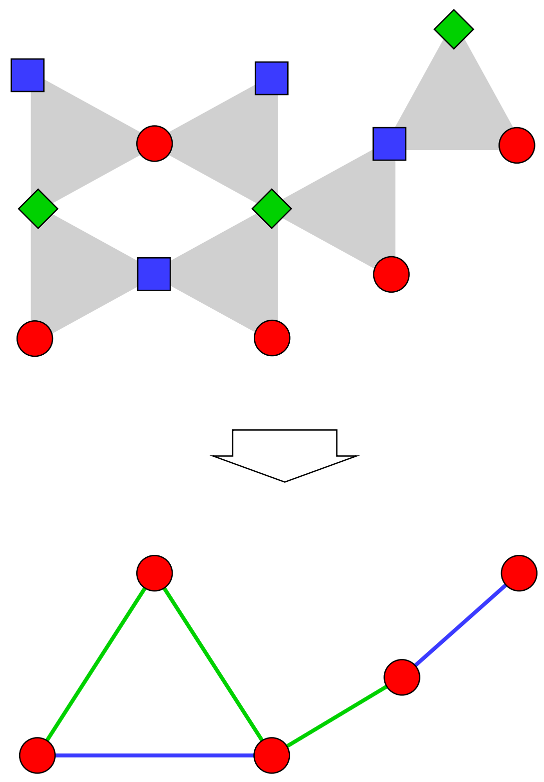

Figure 3.2: Ways of projecting a tripartite graph onto one of its vertex types (red in this case). Red vertices in the projected graph can be connected if they share a green neighbor (green edges in the projected graph), a blue neighbor (blue edges), or a neighbor of either kind (all edges together).

of red and green vertices, in which a red and a green vertex are connected by an ordinary edge if they were connected by a hyperedge in the original network. Thus one can create a network of, for example, photos and the tags applied to them, while dropping information about which users applied which tags. And again one can also construct the equivalent projections onto red/blue and blue/green vertex combinations. Alternatively, one can construct a red/green network by connecting any pair of vertices—of different colors or not—if they share a common blue neighbor. Thus a tag would be connected to a photo if any user applied that tag to that photo, but tags would also be connected to other tags that were used by the same user.

Many other standard concepts in the theory of networks can be generalized to tripartite graphs, including clustering coefficients, correlations between the degrees of adjacent vertices (including three-point correlations), community structure and modularity, motif counts, and more. The concepts introduced above, however, will be sufficient for what is to follow in this chapter.

## 3.3 Random tripartite graphs

In theoretical studies of networks, random graph models have received particular emphasis because they capture many of the essential properties of networked systems in the real world while simultaneously being amenable to analytic treatment. A variety of random graph models have been studied, from models of simple undirected or directed graphs to more complicated examples with correlations, communities, or bipartite structure [57, 102, 10, 117, 107]. In this section we develop the theory of random tripartite hypergraphs with given degree distributions, which turn out to model many of the properties of real tripartite graphs quite effectively. Random hypergraphs have received some attention previously within the mathematics community [37, 59, 137], particularly in the context of combinatorical problems such as graph coloring, and some general results are known concerning the component structure [138, 42, 47]. Here we concentrate more narrowly on results relevant to our primary interest in tripartite graphs.

### 3.3.1 The model

Consider a model hypergraph with $n_r$ red vertices, $n_g$ green vertices, and $n_b$ blue vertices. Each vertex is assigned a degree, corresponding to the number of hyperedges it will have. These degrees can be visualized as "stubs" of hyperedges emerging from each vertex in the appropriate numbers. The degrees must satisfy Eq. (3.2), so that the total number of stubs emerging from vertices of each color is the same and equal to the total desired number of hyperedges $m$.

A total of $m$ three-way hyperedges are now created by choosing trios of stubs uniformly at random, one each from a red, green, and blue vertex, and connecting them to form hyperedges. This model is the equivalent for our tripartite graph of configuration model for unipartite graphs [102] and the random bipartite graph model of [117] for bipartite graphs.

Given the definition of the model, we can, for example, calculate the probability that a hyperedge exists between a given trio of vertices $i, j, k$. In the process of creating a single hyperedge, the probability that we will choose a specific stub attached to red vertex $i$ is $1/m$, since there are a total of $m$ stubs attached to red vertices and we choose uniformly among them. If $i$ has degree $k_i$ then the total probability of choosing a stub from vertex $i$ is $k_i/m$. Similarly the probability of choosing stubs from green and blue vertices $j$ and $k$ are $k_j/m$ and $k_k/m$. Given that there are $m$ hyperedges in total, the overall probability of a hyperedge between $i$, $j$, and $k$ is then

$$P_{ijk} = m \times \frac{k_i}{m} \times \frac{k_j}{m} \times \frac{k_k}{m} = \frac{k_i k_j k_k}{m^2}. \tag{3.5}$$

Via a similar argument, the probability that there is a hyperedge connecting a particular red/green pair $i, j$ (or any other color combination) is $k_i k_j/m$. Note that in a sparse graph in which the typical degrees remain constant as the size of the graph increases, both of these probabilities vanish as $1/m$. Among other things, this implies that the chance of occurrence of small loops in the network vanishes in the limit of large graph size. In the language of graph theory, one says that the network is *locally tree-like*, a property that will be important in the developments to follow.

Rather than specifying the degree of every vertex in the network, we can alternatively specify just the degree distributions $p_r(k)$, $p_g(k)$, and $p_b(k)$ of the three vertex types (constrained to satisfy the sum rules (3.3) and (3.4)), then draw a specific sequence of degrees from those distributions and connect the vertices as before. As a practical matter, if one wanted to generate actual example networks on a computer, one would need to ensure that the degrees satisfied Eq. (3.2), which in general they will not on first being drawn from the distributions. A simple strategy for ensuring that they do is first to draw a complete set of degrees and then repeatedly choose at random a trio of vertices, one of each color, discard the current values of their degrees, and redraw them from the appropriate distributions until the constraint is satisfied.

The degree distributions represent the probability that a vertex of a given color chosen at random from the entire network has a given degree. If we choose a *hyperedge* at random, however, and follow it to the red, green, or blue vertex at one of its corners, that vertex will not have degree distributed according to $p_r(k)$, $p_g(k)$, or $p_b(k)$, and the reason is easy to see: vertices with many hyperedges are proportionately more likely to be encountered when following edges. A vertex of degree ten, for instance, has ten times as many chances to be chosen in this way than a similarly colored vertex of degree one. (And a vertex of degree zero will never be chosen at all.) Thus the distribution of degrees of vertices encountered is proportional to $kp_r(k)$ for red vertices, and similarly for green and blue. Requiring this distribution to sum to unity, the correctly normalized distribution is $kp_r(k)/\sum_k kp_r(k) = kp_r(k)/c_r$.

As in other random graph models, we are in fact usually interested not in the degree of the vertex we encounter but in the number of hyperedges attached to it other than the one we followed to reach it. This is the hypergraph analog of the excess degree, which is 1 less than the total degree, has the same distribution as above, but with the replacement $k \to k+1$, giving an *excess degree distribution* of

$$q_r(k) = \frac{(k+1)p_r(k+1)}{c_r}, \tag{3.6}$$

and similarly for other vertex colors.

### 3.3.2 Generating functions

The fundamental tools we will use in calculating the properties of the random tripartite graph are probability generating functions. We begin by defining generating functions for the degree distributions thus:

$$r_0(z) = \sum_{k=0}^{\infty} p_r(k) z^k, \tag{3.7a}$$

$$g_0(z) = \sum_{k=0}^{\infty} p_g(k) z^k, \tag{3.7b}$$

$$b_0(z) = \sum_{k=0}^{\infty} p_b(k) z^k. \tag{3.7c}$$

Given these generating functions we can, for instance, easily calculate the means of the distributions: $c_r = r_0'(1)$ and so forth. Higher moments are also straightforward.

We also define corresponding generating functions for the excess degree distributions:

$$r_1(z) = \sum_{k=0}^{\infty} q_r(k) z^k = \frac{1}{c_r} \sum_{k=0}^{\infty} (k+1) p_r(k+1) z^k = \frac{r_0'(z)}{r_0'(1)}, \tag{3.8a}$$

and

$$g_1(z) = \sum_{k=0}^{\infty} q_g(k) z^k = \frac{g_0'(z)}{g_0'(1)}, \tag{3.8b}$$

$$b_1(z) = \sum_{k=0}^{\infty} q_b(k) z^k = \frac{b_0'(z)}{b_0'(1)}. \tag{3.8c}$$

### 3.3.3 Projections

As a first example, we use our generating functions to calculate the degree distribution for the projection of a tripartite random graph onto one of its vertex types, as described in Section 3.2. Consider first the projection onto (say) red vertices in which two red vertices are joined by an edge if they share a green neighbor. (The blue vertices are ignored in this projection.)

Suppose a given red vertex A has $s$ green neighbors and each of those green neighbors has $t$ red neighbors other than vertex A. Given that $s$ is distributed according to $p_r(s)$ and $t$ is distributed according to $q_g(t)$, the probability $\rho_g(k)$ that A has exactly $k$ neighbors in the projected network is

$$\rho_g(k) = \sum_{s=0}^{\infty} p_r(s) \sum_{t_1=0}^{\infty} q_g(t_1) \dots \sum_{t_s=0}^{\infty} q_g(t_s)\, \delta\Big(k, \sum_{n=1}^{s} t_n\Big), \tag{3.9}$$

where $\delta(i,j)$ is the Kronecker delta. Multiplying both sides by $z^k$ and summing over $k$, the generating function for this probability distribution is,

$$\begin{aligned}
R_g(z) &= \sum_{k=0}^{\infty} z^k \sum_{s=0}^{\infty} p_r(s) \\
&\qquad \times \sum_{t_1=0}^{\infty} q_g(t_1) \dots \sum_{t_s=0}^{\infty} q_g(t_s)\, \delta\Big(k, \sum_{n=1}^{s} t_n\Big) \\
&= \sum_{s=0}^{\infty} p_r(s) \sum_{t_1=0}^{\infty} q_g(t_1) \dots \sum_{t_s=0}^{\infty} q_g(t_s) z^{\sum_n t_n} \\
&= \sum_{s=0}^{\infty} p_r(s) \sum_{t_1=0}^{\infty} q_g(t_1) z^{t_1} \dots \sum_{t_s=0}^{\infty} q_g(t_s) z^{t_s} \\
&= \sum_{s=0}^{\infty} p_r(s) \left[ \sum_{t=0}^{\infty} q_g(t) z^t \right]^s = \sum_{s=0}^{\infty} p_r(s) \big[g_1(z)\big]^s \\
&= r_0(g_1(z)). \qquad (3.10)
\end{aligned}$$

We can also calculate the generating function for the projection in which two red vertices are connected by an edge if they share either a green or a blue neighbor. The probability for a vertex to have $k$ neighbors in this network is

$$\begin{aligned}
\rho_{gb}(k) &= \sum_{s=0}^{\infty} p_r(s) \sum_{t_1=0}^{\infty} q_g(t_1) \dots \sum_{t_s=0}^{\infty} q_g(t_s) \\
&\qquad \times \sum_{u_1=0}^{\infty} q_b(u_1) \dots \sum_{u_s=0}^{\infty} q_b(u_s)\, \delta\Big(k, \sum_{n=1}^{s} (t_n + u_n)\Big), \qquad (3.11)
\end{aligned}$$

and the corresponding generating function is

$$\begin{aligned}
R_{gb}(z) &= \sum_{k=0}^{\infty} z^k \sum_{s=0}^{\infty} p_r(s) \sum_{t_1=0}^{\infty} q_g(t_1) \ldots \sum_{t_s=0}^{\infty} q_g(t_s) \\
&\quad \times \sum_{u_1=0}^{\infty} q_b(u_1) \ldots \sum_{u_s=0}^{\infty} q_b(u_s)\, \delta\Big(k, \sum_{n=1}^{s} (t_n + u_n)\Big) \\
&= \sum_{s=0}^{\infty} p_r(s) \left[\sum_{t=0}^{\infty} q_g(t) z^t\right]^s \left[\sum_{u=0}^{\infty} q_b(u) z^u\right]^s \\
&= r_0(g_1(z) b_1(z)). \qquad (3.12)
\end{aligned}$$

We can use this result to calculate, for instance, the average degree in the projected network, which is given by

$$R'_{gb}(1) = r'_0(1)\big[b'_1(1) + g'_1(1)\big]. \qquad (3.13)$$

We will also use it in Section 3.4 to compare predictions of the random graph model with real-world networks.

### 3.3.4 Formation and size of the giant component

In this section we examine the component structure of our model network, focusing on the giant component. As with all networks, if our tripartite network is sufficiently sparse—if it has very few edges for the given number of vertices—then vertices will be connected together only in small groups or *small components.* If, however, the number of edges is sufficiently high, then a fraction of the vertices will join together into a single large group, the *giant component*, with the remainder in small components. There is a phase transition with increasing density at which the giant component forms that is closely analogous to the phase transition in classical percolation.

There is more than one possible definition of a component in our tripartite network, but the simplest approach is to define it as a set of vertices of any colors that are connected via hyperedges such that every vertex in the set is reachable from every other by some path through the network. Thus the collection of vertices depicted in the top panel of Fig. 3.2 constitutes a component in this sense.

When viewed in the context of folksonomies, components, and particularly the giant component, play an important practical role. In a folksonomy such as that of Flickr, the photography web site, users can “surf” between photographs by traversing the hypergraph. A user can, for example, click on the tag associated with a photo and see a list of other photos with the same tag. Similarly a user can click on the name of another user and see a list of photos that user has tagged. The existence, or not, of a giant component in the network dictates whether this type of surfing is actually useful or not. If there is no giant component, then surfing users will find themselves restricted to the small set of photos, tags, and users in the component in which they start their surfing. But if there is a giant component then users will be able to surf to a significant fraction of all photos on the entire web site just by clicking on tags or users that seem interesting. The same considerations affect automated surfing by computerized “crawlers” that crawl web sites either to perform directed searches (so-called “spiders”) or to create indexes for later search. If there is no giant component in the folksonomy, then it cannot be crawled in a useful way.

We can calculate properties of the giant component in our tripartite random graph by methods similar to those used for ordinary random graphs [117]. Consider a randomly chosen hyperedge in the full hypergraph, as depicted in Fig. 3.3, and let us calculate the probability that this hyperedge is *not* a part of the giant component. We define $u_r$ to be the probability that the hyperedge is not connected to the giant component via its red vertex, and similarly for $u_g$ and $u_b$, so that the total probability of not belonging to the giant component is $u_r u_g u_b$.

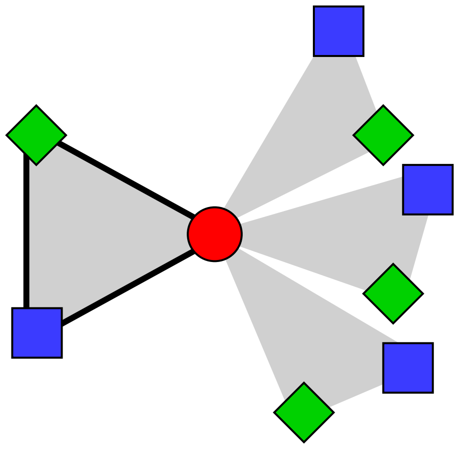

Figure 3.3: If a hyperedge (outlined in bold) is not to belong to the giant component, then it must be that none of the hyperedges reachable via, for instance, its red vertex are themselves members of the giant component.

Suppose that the excess degree of the red vertex—the number of other hyperedges attached to it—is $k$. (In the example shown in Fig. 3.3 we have $k = 3$.) In order that the hyperedge be not connected to the giant component via the red vertex it must be that none of these other hyperedges are connected to the giant component either. Any one hyperedge satisfies this criterion with probability $u_g u_b$—the probability that neither of its other corners lead to the giant component—and all $k$ of them together do so with probability $(u_g u_b)^k$.

The excess degree is distributed according to the distribution $q_r(k)$ defined in Eq. (3.6). Averaging over this distribution, we then derive an expression for $u_r$ thus:

$$u_r = \sum_{k=0}^{\infty} q_r(k)(u_g u_b)^k = r_1(u_g u_b). \tag{3.14}$$

Similarly we can show that

$$u_g = g_1(u_b u_r), \qquad u_b = b_1(u_r u_g). \tag{3.15}$$

The simultaneous solution of these three equations for $u_r$, $u_g$, and $u_b$ then allows us to calculate the probability $1 - u_r u_g u_b$ that a randomly chosen hyperedge is in the giant component. Alternatively, the probability that a randomly chosen red vertex is not in the giant component is the probability that none of its $k$ hyperedges lead to the giant component, which is $\sum_k p_r(k)(u_g u_b)^k = r_0(u_g u_b)$, so the that a red vertex *is* in the giant component with probability

$$S_r = 1 - r_0(u_g u_b), \tag{3.16}$$

and we can write similar equations for $S_g$ and $S_b$. $S_r$ can also be thought of as the fraction of red vertices in the giant component, and hence is a measure of the size of that component. The absolute number of red vertices in the giant component is $n_r S_r$ and the number of vertices of all colors is $n_r S_r + n_g S_g + n_b S_b$.

As in other random graph models, it is in most cases not possible to solve Eqs. (3.14) and (3.15) for $u_r$, $u_g$, and $u_b$ in closed form, but a numerical solution can be found easily by iteration starting from suitable initial values.

We can also derive a condition for the existence of a giant component in the network. A giant component exists if and only if $u_r$, $u_g$, and $u_b$ are all less than 1. (They must all be less than 1 because an extensive giant component of vertices of any one color automatically implies an extensive component of the other two colors, since, with only mild conditions on the degree distribution, the first color must be connected into a giant component by an extensive number of hyperedges, and each hyperedge is attached to one vertex of each color.)

Consider values of the variables that are only slightly different from 1 thus:

$$u_r = 1 - \epsilon_r, \qquad u_g = 1 - \epsilon_g, \qquad u_b = 1 - \epsilon_b, \tag{3.17}$$

where $\epsilon_r$, $\epsilon_g$, and $\epsilon_b$ are small. Then, from Eq. (3.14),

$$\begin{aligned}\epsilon_r = 1 - u_r = 1 - r_1(u_g u_b) &= 1 - r_1(1 - \epsilon_g - \epsilon_b + \epsilon_g \epsilon_b) \\ &= (\epsilon_g + \epsilon_b) r_1'(1) + \mathrm{O}(\epsilon^2), \end{aligned} \tag{3.18}$$

where we have performed a Taylor expansion of $r_1$ and made use of $r_1(1) = 1$ (which is necessarily true if $q_r(k)$ is a properly normalized distribution). We can derive similar equations for $\epsilon_g$ and $\epsilon_b$ and combine all three into the single vector equation

$$\begin{pmatrix} \epsilon_r \\ \epsilon_g \\ \epsilon_b \end{pmatrix} = \begin{pmatrix} 0 & r & r \\ g & 0 & g \\ b & b & 0 \end{pmatrix} \begin{pmatrix} \epsilon_r \\ \epsilon_g \\ \epsilon_b \end{pmatrix}, \tag{3.19}$$

where we have introduced the shorthand $r = r_1'(1)$, $g = g_1'(1)$, and $b = b_1'(1)$.

If $u_r$, $u_g$, and $u_b$ are to be less than 1, meaning the corresponding $\epsilon$'s must all be non-zero, then this equation implies the determinant condition

$$\begin{vmatrix} -1 & r & r \\ g & -1 & g \\ b & b & -1 \end{vmatrix} = 0, \tag{3.20}$$

or

$$2rgb + rg + gb + br = 1. \tag{3.21}$$

This condition defines the point at which the phase transition takes place. Equivalently, $2rgb + rg + gb + br$ crosses 1 at the transition. In fact it is greater than 1 when there is a giant component and less 1 when there is none (rather than the other way around) as can be shown by exhibiting any example where this is the case. A suitable example is provided by a network in which all vertices have degree one, which clearly has no giant component. This choice makes $r = g = b = 0$ and the result follows.

Thus our condition for the existence of a giant component is,

$$2rgb + rg + gb + br > 1. \tag{3.22}$$

This is the equivalent of the well known condition of Molloy and Reed for the existence of a giant component in a unipartite random graph [102].

An alternative form for this condition can be derived by making use of Eqs. (3.6) and (3.8) to write

$$\begin{aligned} r = r_1'(1) &= \sum_{k=0}^{\infty} k q_r(k) = \frac{1}{c_r}\sum_{k=0}^{\infty} k(k+1)p_r(k+1) \\ &= \frac{1}{c_r}\sum_{k=0}^{\infty} k(k-1)p_r(k) = \frac{\langle k^2\rangle_r}{\langle k\rangle_r} - 1, \end{aligned} \tag{3.23}$$

and similarly for $g$ and $b$. Here $\langle \ldots \rangle_r$ indicates an average over the degree distribution of the red vertices and $c_r = \langle k\rangle_r$.

Substituting these expressions into (3.22), we find, after some algebra, that

$$\frac{\langle k\rangle_r}{\langle k^2\rangle_r} + \frac{\langle k\rangle_g}{\langle k^2\rangle_g} + \frac{\langle k\rangle_b}{\langle k^2\rangle_b} < 2. \tag{3.24}$$

This form is particularly pleasing, since it has the same general shape as the criterion of Molloy and Reed for the unipartite case, which can be written as $\langle k\rangle/\langle k^2\rangle < \frac{1}{2}$.

### 3.3.5 Other types of components

The definition of a component used in the previous section is not the only one possible for our tripartite graph. In some folksonomies one cannot surf over connections formed by both users and tags. In some cases, for instance, one is barred from seeing which resources a particular user has tagged for privacy reasons, meaning one can surf between resources with the same tag, but not with the same user. In this case we are surfing on the network formed by two colors of vertices only, say red and green.

We can approach this situation using the same techniques as in the previous section. We define probabilities $u_r$ and $u_g$ as before and find that they satisfy the equations,

$$u_r = r_1(u_g), \qquad u_g = g_1(u_r). \tag{3.25}$$

Linearizing around the point $u_r = u_g = 1$ we then find that the transition at which the giant component appears takes place when

$$\begin{vmatrix} -1 & r \\ g & -1 \end{vmatrix} = 0, \tag{3.26}$$

or equivalently $rg = 1$, with $r$ and $g$ defined as before. By considering appropriate special cases, one can then show that the giant component exists if and only if $rg > 1$. Substituting from Eq. (3.23), we can also write this condition in the form,

$$\frac{\langle k\rangle_r}{\langle k^2\rangle_r} + \frac{\langle k\rangle_g}{\langle k^2\rangle_g} < 1. \tag{3.27}$$

Note that this expression is not symmetric with respect to permutations of the three color indices, as Eq. (3.24) was. This means that in general giant components for different color pairs will appear at different transitions, and it is possible to have a giant component for one pair without having a giant component for another. Thus for instance in our Flickr example one might be able to surf the network of photos and tags, but not the network of photos and users. (Actually, one can surf both just fine in the real Flickr network.)

### 3.3.6 Percolation

One can also consider percolation processes on tripartite networks. If some vertices are removed from the network then the remaining network may or may not percolate, i.e., possess a giant component. For example, on the Flickr web site users can designate photos as publicly viewable or not, and those that are not are, for all intents and purposes, removed from the network. One cannot use them, for instance, for surfing across the network. There are many ways in which vertices might be removed, but as a simple example let us assume that vertices of only one kind are removed and make the standard percolation assumption that they are removed uniformly at random. (More complicated percolation schemes are certainly possible, with more than one type of vertex removed, different probabilities of removal for different types, or nonuniform removal, and all of these schemes can be studied by methods similar to those outlined here.)

Suppose a fraction $\phi$ of the red vertices in our network are present (or functional) and $1-\phi$ are removed (or nonfunctional). In the language of percolation theory, a fraction $\phi$ of the vertices are occupied. Then define $u_r$ as before to be the probability that the red vertex attached to a random hyperedge does not belong to the giant component, or the *giant cluster* as it is more commonly called in the percolation context. There are two different ways in which this can happen. If the vertex itself has been removed, then it does not belong to the giant cluster. Alternatively, it may be present but, as before, none of its neighbors, either blue or green, are in the giant cluster. This allows us to write down an expression for $u_r$ thus:

$$u_r = 1 - \phi + \phi r_1(u_g u_b). \tag{3.28}$$

The corresponding expressions for $u_g$ and $u_b$ are the same as in our previous calculation, $u_g = g_1(u_b u_r)$, $u_b = b_1(u_r u_g)$, and the fractions of red, green, and blue vertices in the giant percolation cluster are

$$S_r = \phi[1 - r_0(u_g u_b)], \tag{3.29a}$$

$$S_g = 1 - g_0(u_b u_r), \tag{3.29b}$$

$$S_b = 1 - b_0(u_r u_g). \tag{3.29c}$$

We can also calculate an expression for the value of $\phi$ at which the percolation transition happens. As before we perturb around the point $u_r = u_g = u_b = 1$ that corresponds to no giant cluster and the equivalent of Eq. (3.19) is

$$\begin{pmatrix} \epsilon_r \\ \epsilon_g \\ \epsilon_b \end{pmatrix} = \begin{pmatrix} 0 & \phi r & \phi r \\ g & 0 & g \\ b & b & 0 \end{pmatrix} \begin{pmatrix} \epsilon_r \\ \epsilon_g \\ \epsilon_b \end{pmatrix}, \tag{3.30}$$

with $r$, $g$, and $b$ defined as before. This implies that the transition happens at $\phi = \phi_c$ where $\phi_c$ is the solution of $2\phi rgb + \phi rg + gb + \phi br = 1$. That is,

$$\phi_c = \frac{1 - gb}{r(2gb + g + b)}. \tag{3.31}$$

Making use of Eq. (3.23) and the corresponding expressions for $g$ and $b$ we then find that

$$\phi_c = \left( \frac{\langle k^2 \rangle_r}{\langle k \rangle_r} - 1 \right)^{-1} \left[ \left( 2 - \frac{\langle k \rangle_g}{\langle k^2 \rangle_g} - \frac{\langle k \rangle_b}{\langle k^2 \rangle_b} \right)^{-1} - 1 \right]. \tag{3.32}$$

### 3.3.7 Simulations

Before looking at real-world tripartite networks, we first compare our calculations with simulation results for computer-generated random graphs.

Consider a tripartite random graph with Poisson degree distributions thus:

$$p_r(k) = \mathrm{e}^{-c_r} \frac{c_r^k}{k!}, \quad p_g(k) = \mathrm{e}^{-c_g} \frac{c_g^k}{k!}, \quad p_b(k) = \mathrm{e}^{-c_b} \frac{c_b^k}{k!}, \tag{3.33}$$

where the average degrees $c_r$, $c_g$, and $c_b$ satisfy Eq. (3.2). The corresponding generating functions are

$$\begin{aligned} r_0(z) = r_1(z) &= \mathrm{e}^{-c_r} \sum_{k=0}^{\infty} \frac{c_r^k}{k!} z^k = \mathrm{e}^{c_r(z-1)}, \\ g_0(z) = g_1(z) &= \mathrm{e}^{-c_g} \sum_{k=0}^{\infty} \frac{c_g^k}{k!} z^k = \mathrm{e}^{c_g(z-1)}, \\ b_0(z) = b_1(z) &= \mathrm{e}^{-c_b} \sum_{k=0}^{\infty} \frac{c_b^k}{k!} z^k = \mathrm{e}^{c_b(z-1)}. \end{aligned} \tag{3.34}$$

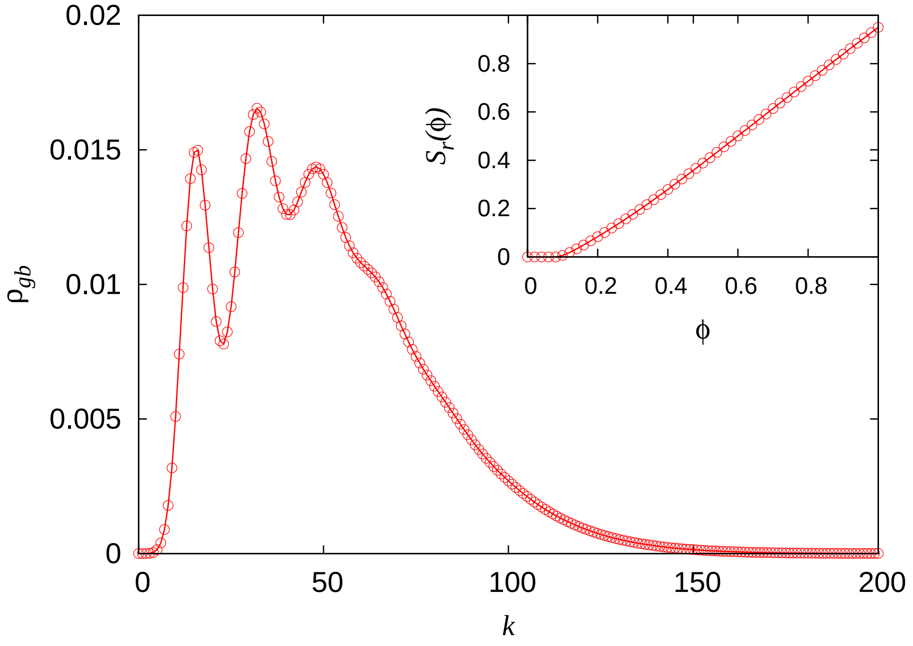

Figure 3.4: The degree distribution for the projection of our Poisson hypergraph onto its red vertices alone, in which two red vertices are joined by an edge if they have either a green or a blue neighbor in common on the original tripartite network. The solid line is the exact solution, Eq. (3.36), and the points are the results of numerical simulations averaged over a hundred realizations of the network. The error bars are smaller than the size of the points in all cases. Inset: The fraction of red vertices belonging to the giant percolation cluster for site percolation on the tripartite network, as a function of occupation probability $\phi$. The solid line is the exact solution and the points are the results of numerical simulations.

We can use these to calculate, for instance, the degree distribution of the projection of the network onto the red vertices in which two vertices are connected if they share either a green or a blue neighbor. The generating function for this distribution is given by Eq. (3.12) to be

$$R_{gb} = r_0(g_1(z)b_1(z)) = \mathrm{e}^{c_r(\mathrm{e}^{(c_g+c_b)(z-1)}-1)}. \tag{3.35}$$

Expanding in powers of $z$, we then find that the probability $\rho_{gb}(k)$ of a red vertex having exactly $k$ neighbors in the projected network is

$$\rho_{gb}(k) = \frac{(c_g+c_b)^k}{k!}\mathrm{e}^{c_r(\mathrm{e}^{-(c_g+c_b)}-1)} \sum_{m=1}^{k} \left\{ {k \atop m} \right\} \left[c_r \mathrm{e}^{-(c_g+c_b)}\right]^m, \tag{3.36}$$

where $\left\{ {k \atop m} \right\}$ is a Stirling number of the second kind, i.e., the number of ways of dividing $k$ objects into $m$ nonempty sets [1].

The main panel of Fig. 3.4 shows the form of this distribution for the case $c_r = 3$, $c_g = 10$, $c_b = 6$. In the same plot we show the results of simulations in which random tripartite graphs with the same degree distributions and $n_r = 100\,000$, $n_g = 30\,000$, and $n_b = 50\,000$ were generated and then explicity projected onto the red vertices and the resulting degree distribution measured directly. As the figure shows, the agreement between the two is excellent.

The inset of Fig. 3.4 shows the size of the giant cluster for percolation on the red vertices of the same network as a function of the occupation probability $\phi$, calculated both by numerical solution of Eqs. (3.28)–(3.29) and by direct measurement on simulated networks. Again the agreement is excellent.

## 3.4 Comparison with real-world data

In this section we compare the predictions of our tripartite random graph model against data for the folksonomy of the Flickr photo-sharing web site. As we show, the theory and empirical observations agree well in some respects, but less well in others. In many ways the discrepancies are at least as interesting as the cases of agreement, since they indicate situations in which the structure of the observed network cannot be explained by a simple random model that ignores social and other effects. When data and model disagree it is a sign that these effects are important in determining the network structure. Thus, as with other random graph models, one of the most significant roles our model can play may be as a null model that allows the experimenter to determine when a network is doing something nontrivial.

Our example data set represents the folksonomy network of 266 198 photos added to the Flickr web site by its users during 2007, along with the tags applied to those

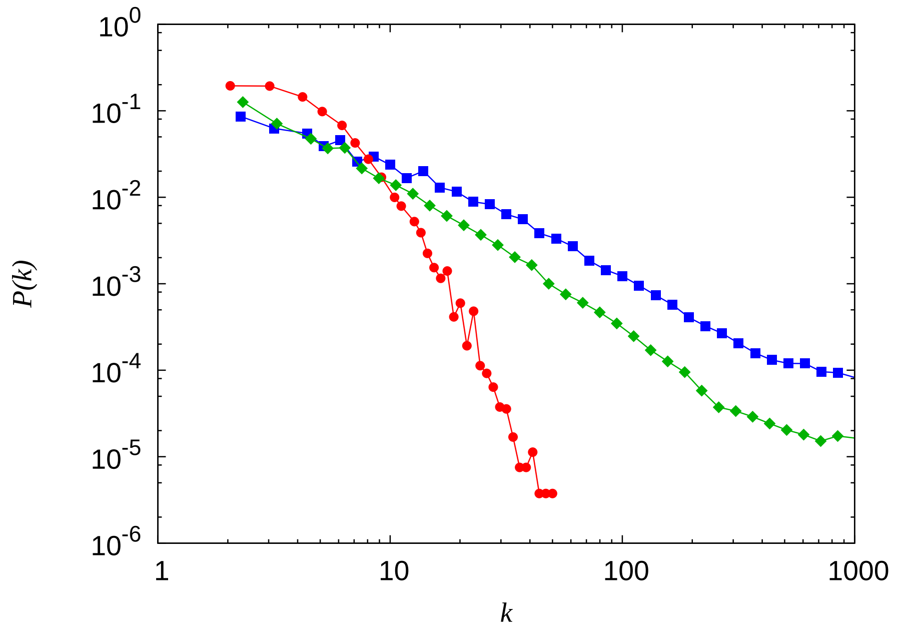

Figure 3.5: The three degree distributions of the tripartite Flickr folksonomy network for photos (red), tags (green), and users (blue).

photos and the users who applied them. The first step in analyzing the data is to measure the three degree distributions for the three types of vertices. The degree distributions are shown in Fig. 3.5. As is common in most social networks, they are highly right-skewed, meaning there are many vertices of low degree and a small number of very high degree, although the distributions do not follow power-law forms as the distributions in some networks do. Using these distributions, we can, following Eqs. (3.7) and (3.8), construct the corresponding generating functions, which are simple polynomials (albeit of high order) that can be easily evaluated numerically.

We can use our generating functions to calculate, for example, the generating functions $R_{gb}(z)$ and so forth for the degree distributions of the projections of the network onto one vertex type, using Eqs. (3.10) and (3.12) and their equivalents for other vertex types. Again these functions can be rapidly evaluated for any argument $z$ numerically. The degree distributions themselves are then given by derivatives of the generating functions thus:

$$p_k = \frac{1}{k!} \frac{\mathrm{d}^k R_{gb}}{\mathrm{d}z^k}\bigg|_{z=0}. \tag{3.37}$$

Direct numerical evaluation of derivatives is plagued by problems with noise and should be avoided, but one can get good results [104] by instead employing Cauchy's integral formula for the $k$th derivative of a function:

$$\frac{\mathrm{d}^k f}{\mathrm{d}z^k}\bigg|_{z=z_0} = \frac{k!}{2\pi\mathrm{i}} \oint \frac{f(z)}{(z-z_0)^{k+1}}\,\mathrm{d}z, \tag{3.38}$$

where the integral is around a contour enclosing the point $z_0$ but excluding any poles of $f(z)$. Applying this formula to (3.37) we get

$$p_k = \frac{1}{2\pi\mathrm{i}} \oint \frac{R_{gb}(z)}{z^{k+1}}\,\mathrm{d}z. \tag{3.39}$$

We then calculate the degree distribution by performing the contour integral numerically around a suitable contour (the unit circle $|z| = 1$ works well). One can without difficulty calculate to good precision the first thousand or so coefficients of the generating function in this fashion.

We have performed this calculation using the degree distributions of the Flickr network and projecting onto the resources, i.e., the photos. Figure 3.6 shows a comparison of the results with the degree distribution for the actual projected network. The upper solid line in the figure represents the theoretical result, while the circles represent the measurements. Although the two curves have the same general shape, it's clear from the figure that the agreement between them is only moderately good in this case. Upon closer inspection, however, it turns out that there is a relatively simple reason for this.

As discussed in Section 3.3.1, our random graph model assumes a locally-tree like structure for the tripartite network, a structure with no short loops. The Flickr network, on the other hand, turns out to have many short loops, which is why empirical measurements and model do not agree in Fig. 3.6. As we now show, however, the loops in the Flickr network are primarily of a trivial kind that can easily be allowed for in the calculations.

Typically, photos are not added to the Flickr network individually, but in sets. The most common practice is for a user to upload a set of photos on a particular subject—say, pictures of a Ferrari motor car—and then label all of the photos in the set with the same set of tags—*Ferrari*, *automobile*, *sports car*, and so forth. This creates short loops between photos in the set of the form $P_1 \to T_1 \to P_2 \to T_2 \to P_1$, where the $P$s are the photos and the $T$s are tags. These loops will have an adverse affect on the calculation of the number of neighbors a photo has in the projected network, since in many cases two projected edges from a photo will lead to the same neighboring photo, rather than to different neighbors, and hence give a lower degree in the projected network than our naive random graph calculation.

To test the effect of these "trivial" loops in the network structure, we have pruned the data set to remove instances of multiple tagging. In the pruned data set the application by a user of many tags to the same photo is represented by just a single hyperedge, rather than many. In this representation, hyperedges represent the act of tagging a photo, rather than a specific tag, and only one hyperedge is included between a user and a photo no matter how many tags the user applies. Similarly we also represent the tagging of many photos with the same tag by a single hyperedge, so that hyperedges represent the act of tagging an entire photo set, rather than just a single photo. This should remove most instances of trivial loops in the projected network of the type described above. We note, that a similar observation on the effects of multiple tagging was made by [35].

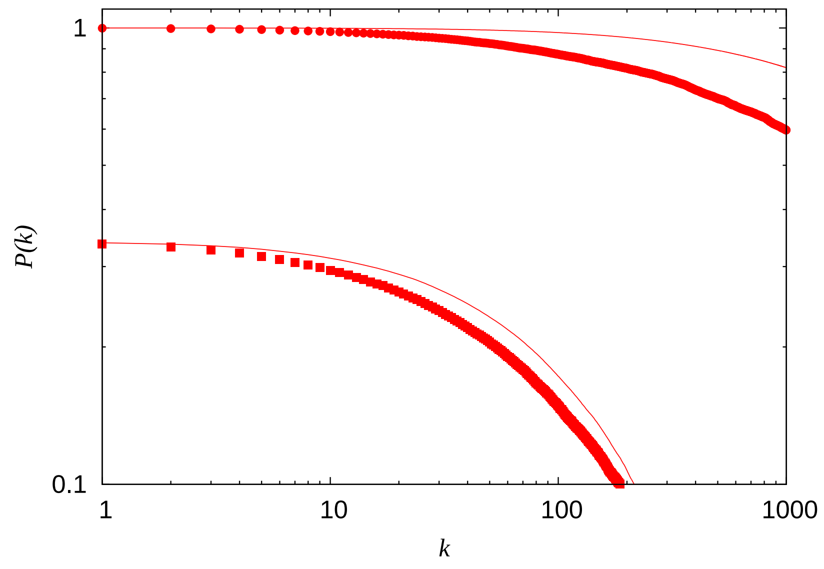

Figure 3.6: Circles show the cumulative distribution function for the degree distribution of the projection of the Flickr network onto its photograph vertices, while the upper solid line shows the predictions of the random graph model for the same quantity. Squares show the same function after pruning of the data to remove multiple tagging as described in the text and the lower solid curve shows the corresponding model prediction, recalculated from the new degree distributions after pruning.

Now we calculate again the projection of the hypergraph onto the set of photos. We also recalculate the theoretical predictions to reflect the changed degree distributions of the hypergraph following pruning. The results are shown in Fig. 3.6 (squares and lower solid curve) and, as the figure shows, the agreement is now quite good between theory and observation. This suggests that the earlier disagreement between the two is indeed primarily a result of the presence of the loops in the hypergraph introduced by the practice of multiple tagging.

We can perform similar calculations for projections onto other types of vertices. In Fig. 3.7 we show degree distributions, before and after pruning of the data set, for the projection onto users. Agreement between theory and observation for the unpruned data is again quite poor in this case but significantly better for the pruned data.

These calculations provide, in many ways, a good example of the utility of random

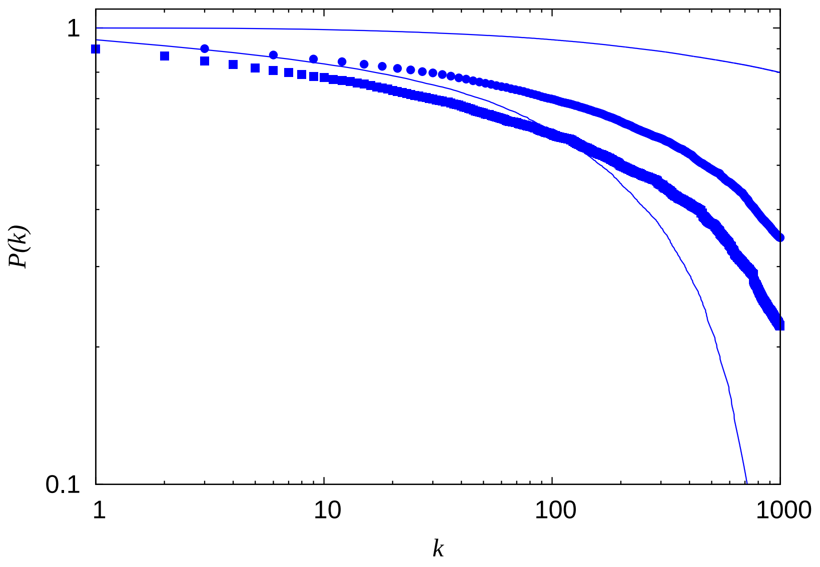

Figure 3.7: Cumulative distribution functions for the degree distributions of the projection of the Flickr network onto its user vertices, both before and after pruning of the data. The points represent the observations, unpruned (circles) and pruned (squares), while the solid lines represent the predictions of the model.

graph models. When compared with the raw data from the Flickr network, our random graph model agrees qualitatively, but not quantitatively, indicating that there are effects present in the network that are not accounted for by simple random hyperedges. On the other hand, once one prunes the data to remove multiple tagging, the agreement becomes much better, suggesting that multiple tagging is the primary nonrandom behavior taking place in the network and that in other respects the network is in fact quite close to being a random graph. Thus the model allows us not only to say when the network deviates from the random assumption, but also the particular nature of the deviation.

## 3.5 Discussion

Motivated by the emergence of new types of social networks, such as folksonomies, we have in this chapter proposed and studied a model of random tripartite hypergraphs. We have defined basic network measures, such as degree distributions and projections onto individual vertex types, and calculated a variety of statistical properties of the model in the limit of large network size. Among other things we have calculated the explicit degree distributions for projected networks, conditions for the emergence of a giant component, the size of the giant component when there is one, and the location of the percolation threshold for site percolation on the network. In principle, the techniques introduced could be extended to hypergraphs with more vertex types or additional types of edges, although we have not pursued any such extensions here.

We have compared our results against measurements of computer-generated random hypergraphs and a real-world tripartite network, the folksonomy of the on-line photo-sharing web site Flickr. In the latter case, we have focused on the degree distributions of projections of the hypergraph onto one vertex type and find that in some instances the theory makes predictions in moderately good agreement with the observations while in others the agreement is poorer. In all cases, however, we find that agreement becomes significantly better when we remove instances of multiple tagging from the network—instances in which a user applies many tags to the same photo or the same tag to many photos—suggesting that the disagreement is primarily a result of relatively trivial structures in the network, rather than more subtle or large-scale social network effects.

CHAPTER IV

# Network growth models I: Equilibrium degree distributions for networks with vertex and edge turnover

## 4.1 Introduction

In the previous two chapters we focussed on the properties of static networks—those in which the number of vertices and edges are fixed—here, we revert to the case of dynamic networks. In particular we will examine the case of growing networks (discussed in Sec. 1.6.1), where the network evolves by the addition/deletion of vertices and edges.

A huge amount of of work has been devoted to the case of growing networks, with citation networks [48, 126] and the worldwide web [7, 91] receiving the most attention. Perhaps the best-known body of work on this topic is that dealing with "preferential attachment" models [49, 13], in which vertices are added to a network with edges that attach to preexisting vertices with probabilities depending on those vertices' degrees—see Sec. 1.6.1 for a detailed description. When the attachment probability is precisely linear in the degree of the target vertex the resulting degree sequence for the network follows a Yule distribution in the limit of large network size, meaning it has a power-law tail [49, 13, 96, 50, 12]. This case is of special interest because both citation networks and the worldwide web are observed to have degree distributions that approximately follow power laws.

The preferential attachment model may be quite a good model for citation networks, which is one of the cases for which it was originally proposed [49, 96]. For other networks, however, and especially for the worldwide web, it is, as many authors have pointed out, necessarily incomplete [50, 5, 95, 143, 71]. On the web there are clearly other processes taking place in addition to the deposition of vertices and edges. In particular, it is a matter of common experience that vertices (i.e., web pages) are often removed from the web, and with them the links that they had to other pages. Models of this process have been touched upon occasionally in the literature [136, 40, 43] and the evidence suggests that in some cases vertex deletion affects the crucial power-law behavior of the degree distribution, while in other cases it does not.

In this chapter, we study the general process in which a network grows (or, potentially, shrinks) by the constant addition and removal of vertices and edges. We show that a class of such processes can be solved exactly for the degree distributions they generate by solving differential equations governing the probability generating functions for those distributions. In particular, we give solutions for three example problems of this type, having uniform or preferential attachment, and having stationary size or net growth. The case of uniform attachment and stationary size is of interest as a possible model for the structure of peer-to-peer filesharing networks (discussed in detail in the following chapter), while the preferential-attachment stationary-size case displays a nontrivial stretched exponential form in the tail of the degree distribution. Our solution of the preferential attachment case with net growth confirms earlier results indicating that this process generates a power-law distribution, although the exponent of the power-law diverges as the growth rate tends to zero, giving degree distributions that are numerically indistinguishable from exponential for small growth rates. This suggests that the clear power law seen in the real world-wide web is a signature of a network whose rate of vertex accrual far outstrips the rate at which vertices are removed. The relative rates of addition and removal could, however, change as the web matures, possibly leading to a loss of power-law behavior at some point in the future.

## 4.2 The model

Consider a network that evolves by the addition and removal of vertices. In each unit of time, we add a single vertex to the network and remove $r$ vertices. When a vertex is removed so too are all the edges incident on that vertex, which means that the degrees of the vertices at the other ends of those edges will decrease. Non-integer values of $r$ are permitted and are interpreted in the usual stochastic fashion. (For example, values $r < 1$ can be interpreted as the probability per unit time that a vertex is removed.) The value $r = 1$ corresponds to a network of fixed size in which there is vertex turnover but no growth; values $r < 1$ correspond to growing networks. In principle one could also look at values $r > 1$, which correspond to shrinking networks, and the methods derived here are applicable to that case. However, we are not aware of any real-world examples of shrinking networks in which the asymptotic degree distribution is of interest, so we will not pursue the shrinking case here.

We make two further assumptions, which have also been made by most previous authors in studying these types of systems: (1) that all vertices added have the same initial degree, which we denote $c$; (2) that the vertices removed are selected uniformly at random from the set of all extant vertices. Note however that we will not assume that the network is uncorrelated (i.e., that it is a random multigraph conditioned on its degree distribution as in the configuration model). In general the networks we consider will have correlations among the degrees of their vertices but our solutions will nonetheless be exact.

Let $p_k$ be the fraction of vertices in the network at a given time that have degree $k$. By definition, $p_k$ has the normalization

$$\sum_{k=0}^{\infty} p_k = 1. \tag{4.1}$$

Our primary goal in this paper will to evaluate exactly the degree distribution $p_k$ for various cases of interest.

Although the form of $p_k$ is, as we will see, highly nontrivial in most cases, the mean degree of a vertex $\langle k\rangle = \sum_{k=0}^{\infty} kp_k$ is easily derived in terms of the parameters $r$ and $c$. The mean number of vertices added to the network per unit time is $1-r$. The mean number of edges removed when a randomly chosen vertex is removed from the network is by definition $\langle k\rangle$. Thus the mean number of edges added to the network per unit time is $c - r\langle k\rangle$. For a graph of $m$ edges and $n$ vertices, the mean degree is $\langle k\rangle = 2m/n$. After time $t$ we have $n = (1 - r)t$ and, assuming that $\langle k\rangle$ has an asymptotically constant value, $m = (c - r\langle k\rangle)t$. Thus

$$\langle k\rangle = 2 \times \frac{c - r\langle k\rangle}{1 - r}, \tag{4.2}$$

or, rearranging,

$$\langle k\rangle = \frac{2c}{1 + r}. \tag{4.3}$$

In the special case $r = 1$ of a constant-size network, this gives $\langle k\rangle = c$, which is clearly the correct answer.

We must also consider how an added vertex chooses the $c$ other vertices to which it attaches. Let us define the attachment kernel $\pi_k$ to be $n$ times the probability that a given edge of a newly added vertex attaches to a given preexisting vertex of degree $k$. The factor of $n$ here is convenient, since it means that the total probability that the given edge attaches to any vertex of degree $k$ is simply $\pi_k p_k$. Since each edge must attach to a vertex of *some* degree, this also immediately implies that the correct normalization for $\pi_k$ is

$$\sum_{k=0}^{\infty} \pi_k p_k = 1. \tag{4.4}$$

For the particular case of $\pi_k \propto k$ and $r < 1$, which we consider in Section 4.3.3, models similar to ours have been studied previously by Sarshar and Roychowdhury [136], Chung and Lu [40], and Cooper, Frieze, and Vera [43]. While these authors did not seek an exact solution, our results on the power-law tail of the degree distribution in this case coincide with theirs.

### 4.2.1 Rate equation

Given these definitions, the evolution of the degree distribution is governed by a rate equation as follows. If there are at total of $n$ vertices in the network at a given time then the number of vertices with degree $k$ is $np_k$. One unit of time later this number is $(n+1-r)p'_k$, where $p'_k$ is the new value of $p_k$. Then

$$(n+1-r)p'_k = np_k + \delta_{kc} + c\pi_{k-1}p_{k-1} - c\pi_k p_k + r(k+1)p_{k+1} - rkp_k - rp_k. \tag{4.5}$$

The term $\delta_{kc}$ (Kronecker delta function) in Eq. (4.5) represents the addition of a vertex of degree $c$ to the network. The terms $c\pi_{k-1}p_{k-1}$ and $-c\pi_k p_k$ describe the flow of vertices from degree $k-1$ to $k$ and from $k$ to $k+1$ as they gain extra edges when newly added vertices attach to them. The terms $(k+1)p_{k+1}$ and $-kp_k$ describe the flow from degree $k+1$ to $k$ and from $k$ to $k-1$ as vertices loose edges when one of their neighbors is removed from the network. And the term $-rp_k$ represents the removal with probability $r$ of a vertex with degree $k$. Contributions from processes in which a vertex gains or loses two or more edges in a single unit of time vanish in the limit of large $n$ and have been neglected.

We will be interested in the asymptotic form of $p_k$ in the limit of large times for a given $\pi_k$. Setting $p'_k = p_k$ in (4.5) gives

$$\delta_{kc} + c\pi_{k-1}p_{k-1} - c\pi_k p_k + r(k+1)p_{k+1} - rkp_k - p_k = 0. \tag{4.6}$$

We can write the solution to (5.2) in terms of generating functions as follows. Let us define

$$F_0(z) = \sum_{k=0}^{\infty} \pi_k p_k z^k, \tag{4.7}$$

$$G_0(z) = \sum_{k=0}^{\infty} p_k z^k. \tag{4.8}$$

Then, upon multiplying both sides of (5.2) by $z^k$ and summing over $k$ (with the convention that $p_{-1} = 0$), we derive a differential equation for $G_0(z)$ thus:

$$r(1-z)\frac{\mathrm{d}G_0}{\mathrm{d}z} - G_0(z) - c(1-z)F_0(z) + z^c = 0. \tag{4.9}$$

Note also that we can easily generalize our model to the case where the degrees of the vertices added are not all identical but are instead drawn at random from some distribution $r_k$. In that case, we simply replace $\delta_{kc}$ in Eq. (5.2) with $r_k$ and $z^c$ in Eq. (4.9) with the generating function $H_0(z) = \sum_k r_k z^k$.

In the following sections we solve Eq. (4.9) for a number of different choices of the attachment kernel $\pi_k$. Note that, since the definitions of both $F_0(z)$ and $G_0(z)$ incorporate the unknown distribution $p_k$, we must in general solve implicitly for $G_0(z)$ in terms of $F_0(z)$. In all of the cases of interest to us here, however, it turns out to be straightforward to derive a explicit equation for $G_0(z)$ as a special case of (4.9).

## 4.3 Solutions for specific cases

In this section we study three specific examples of the class of models defined in the preceding section, namely linear preferential attachment models ($\pi_k \propto k$) for both growing and fixed-size networks, and uniform attachment ($\pi_k$ = constant) for fixed size. As we will see, each of these cases turns out to have interesting features.

### 4.3.1 Uniform attachment and constant size

For the first of our example models we study the case where the size of the network is constant ($r = 1$) and in which each vertex added chooses the $c$ others to which it attaches uniformly at random. This means that $\pi_k$ is constant, independent of $k$, and, combining Eqs. (4.1) and (4.4), we immediately see that the correct normalization for the attachment kernel is $\pi_k = 1$ for all $k$. Then we have $\pi_k p_k = p_k$ so that $F_0(z) = G_0(z)$ in (4.9), which gives

$$\left[c + \frac{1}{1-z}\right] G_0(z) - \frac{\mathrm{d}G_0}{\mathrm{d}z} = \frac{z^c}{1-z}. \tag{4.10}$$

Noting that $(1-z)\mathrm{e}^{-cz}$ is an integrating factor and that $G_0(z)$ must obey the boundary condition $G_0(1) = 1$, we readily determine that

$$\begin{aligned} G_0(z) &= \frac{\mathrm{e}^{cz}}{1-z} \int_z^1 t^c \mathrm{e}^{-ct}\,\mathrm{d}t \\ &= \frac{\mathrm{e}^{cz}}{1-z} c^{-(c+1)} \big[\Gamma(c+1, cz) - \Gamma(c+1, c)\big], \end{aligned} \tag{4.11}$$

where

$$\Gamma(c+1, x) = \int_x^\infty t^c \mathrm{e}^{-t}\,\mathrm{d}t. \tag{4.12}$$

is the incomplete $\Gamma$-function [1].

One can easily check that this gives a mean degree $G_0'(1) = c$, as it must, and that the variance of the degree $G_0''(1) + G_0'(1) - c^2$ is equal to $\frac{2}{3}c$, indicating a tightly peaked degree distribution.

To obtain an explicit expression for the degree distribution, we make use of

$$\Gamma(c+1,x) = \Gamma(c+1)\,\mathrm{e}^{-x}\sum_{m=0}^{c}\frac{x^m}{m!}, \tag{4.13}$$

$$\mathrm{e}^x = \sum_{m=0}^{\infty}\frac{x^m}{m!}, \tag{4.14}$$

$$(1-z)^{-1} = \sum_{k=0}^{\infty} z^k, \tag{4.15}$$

to write

$$G_0(z) = c^{-(c+1)} \times \sum_{k=0}^{\infty} z^k \left[\Gamma(c+1)\sum_{m=0}^{c}\frac{(cz)^m}{m!} - \Gamma(c+1,c)\sum_{m=0}^{\infty}\frac{(cz)^m}{m!}\right].$$

The $z$-dependence in the first term of this expression can be rewritten

$$\begin{aligned}
\sum_{k=0}^{\infty} z^k \sum_{m=0}^{c}\frac{(cz)^m}{m!} &= \sum_{m=0}^{c}\sum_{k=m}^{\infty} z^k \frac{c^m}{m!} \\
&= \sum_{k=0}^{\infty} z^k \sum_{m=0}^{\min(k,c)}\frac{c^m}{m!} \\
&= \mathrm{e}^c \times \sum_{k=0}^{\infty} z^k \frac{\Gamma\big(\min(k,c)+1,c\big)}{\Gamma\big(\min(k,c)+1\big)},
\end{aligned} \tag{4.16}$$

where $\min(k,c)$ denotes the smaller of $k$ and $c$ and we have again employed (4.13). A similar sequence of manipulations leads to an expression for the second term also, thus:

$$\sum_{k=0}^{\infty} z^k \sum_{m=0}^{\infty}\frac{(cz)^m}{m!} = \mathrm{e}^c \times \sum_{k=0}^{\infty} z^k \frac{\Gamma(k+1,c)}{\Gamma(k+1)}. \tag{4.17}$$

Combining these identities with (4.16), it is then a simple matter to read off the term in $G_0(z)$ involving $z^k$, which is by definition our $p_k$. We find two separate expressions for the cases of $k$ above or below $c$:

$$p_k = \frac{\mathrm{e}^c}{c^{c+1}}\big[\Gamma(c+1) - \Gamma(c+1,c)\big]\frac{\Gamma(k+1,c)}{\Gamma(k+1)}, \quad \text{for } k < c, \tag{4.18}$$

and

$$p_k = \frac{\mathrm{e}^c}{c^{c+1}}\Gamma(c+1,c)\left[1 - \frac{\Gamma(k+1,c)}{\Gamma(k+1)}\right], \quad \text{for } k \geq c. \tag{4.19}$$

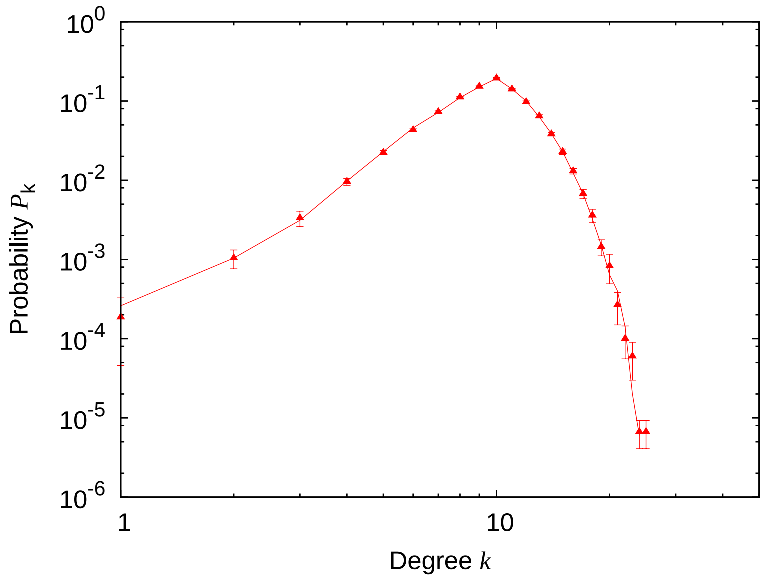

Figure 4.1: The degree distribution of our model for the case of uniform attachment ($\pi_k$ = constant) with fixed size $n = 50\,000$ and $c = 10$. The points represent data from numerical simulations and the solid line is the analytic solution.

Note that the quantity $\Gamma(k+1,c)/\Gamma(k+1)$ appearing in both these expressions is the probability that a Poisson-distributed variable with mean $c$ is less than or equal to $k$. Thus the degree distribution has a tail that decays as the cumulative distribution of such a Poisson variable, implying that it falls off rapidly. To see this more explicitly, we note that for fixed $c$ and $k \gg c$

$$p_k = \frac{\Gamma(c+1,c)}{c^{c+1}} \sum_{m=k+1}^{\infty} \frac{c^m}{m!} \simeq \frac{\Gamma(c+1,c)}{\Gamma(k+2)} c^{k-c}, \tag{4.20}$$

since the sum is strongly dominated in this limit by its first term. Applying Stirling's approximation, $\Gamma(x) \simeq (x/\mathrm{e})^x \sqrt{2\pi/x}$, this gives

$$p_k \simeq \frac{\Gamma(c+1,c)}{c^c} \times k^{-3/2} \mathrm{e}^k \left(\frac{c}{k}\right)^k, \tag{4.21}$$

which decays substantially faster asymptotically than any exponential.

As a check on these calculations, we have performed extensive computer simulations of the model. In Fig. 4.1 we show results for the case $c = 10$, along with the exact solution from Eqs. (4.18) and (4.19). As the figure shows, the agreement between the two is excellent.

Before moving on to other issues, we note a different and particularly simple case of a growing network with uniform attachment, the case in which the vertices added have a Poisson degree distribution $c^k \mathrm{e}^{-c}/k!$ with mean $c$. In that case the factor of $z^c$ in Eq. (4.10) is replaced with the generating function $H_0(z)$ for the Poisson distribution:

$$H_0(z) = \sum_{k=0}^{\infty} \frac{c^k \mathrm{e}^{-c}}{k!} z^k = \mathrm{e}^{c(z-1)}, \tag{4.22}$$

and the solution, Eq. (4.11), becomes

$$G_0(z) = \frac{\mathrm{e}^{cz}}{1-z} \int_z^1 h(t)\, \mathrm{e}^{-ct}\, \mathrm{d}t = \mathrm{e}^{c(z-1)}, \tag{4.23}$$

which is itself the generating function for a Poisson distribution. Thus we see particularly clearly in this case that the equilibrium degree distribution in the steady-state uniform attachment network is sharply peaked with a Poisson tail. In fact, the network in this case is simply an uncorrelated random graph of the type famously studied by Erdős and Rényi [57]. It is straightforward to see that if one starts with such a graph and randomly adds and removes vertices with Poisson distributed degrees, the graph remains an uncorrelated random graph with the same degree distribution, and hence this distribution is necessarily the fixed point of the evolution process, as the solution above demonstrates.

### 4.3.2 Preferential attachment and constant size

Our next example adds an extra degree of complexity to the picture: we consider vertices that attach to others in proportion to their degree, the so-called "preferential attachment" mechanism [13]. This implies that our attachment kernel $\pi_k$ is linear in the degree, $\pi_k = Ak$ for some constant $A$. The normalization requirement (4.4) then implies that

$$\sum_{k=0}^{\infty} \pi_k p_k = A \times \sum_{k=0}^{\infty} k p_k = A\langle k\rangle = 1, \tag{4.24}$$

and hence $A = 1/\langle k\rangle$. For the moment, let us continue to focus on the case $r = 1$ of constant network size, in which case $\langle k\rangle = c$ (Eq. (4.3)) and

$$\pi_k = \frac{k}{c}. \tag{4.25}$$

Then

$$F_0(z) = \frac{1}{c} \times \sum_{k=0}^{\infty} k p_k z^k = \frac{z}{c}\, G_0'(z), \tag{4.26}$$

and Eq. (4.9) becomes

$$\frac{G_0(z)}{(1-z)^2} - \frac{\mathrm{d}G_0}{\mathrm{d}z} = \frac{z^c}{(1-z)^2}. \tag{4.27}$$

The appropriate integrating factor in this case is $\mathrm{e}^{-1/(1-z)}$, which, in conjunction with the boundary condition $G_0(1) = 1$, gives

$$G_0(z) = e^{1/(1-z)} \int_z^1 \frac{t^c}{(1-t)^2} \mathrm{e}^{-1/(1-t)}\, \mathrm{d}t. \tag{4.28}$$

Changing the variable of integration to $y = 1/(1-t)$ this expression can be written

$$\begin{aligned} G_0(z) &= \mathrm{e}^{1/(1-z)} \int_{1/(1-z)}^{\infty} \left(1 - \frac{1}{y}\right)^c \mathrm{e}^{-y}\, \mathrm{d}y \\ &= \mathrm{e}^{1/(1-z)} \sum_{s=0}^{c} (-1)^s \binom{c}{s} \int_{1/(1-z)}^{\infty} \frac{\mathrm{e}^{-y}}{y^s}\, \mathrm{d}y \\ &= 1 + \mathrm{e}^{1/(1-z)} \sum_{s=1}^{c} (-1)^s \binom{c}{s} \Gamma\left(1-s, \frac{1}{1-z}\right). \end{aligned} \tag{4.29}$$

where $\Gamma(1-s, x) = \int_x^{\infty} e^{-y}\, y^{-s}\, \mathrm{d}y$ is again the incomplete $\Gamma$-function, here appearing with a negative first argument.

A useful identify for the case $s \geq 1$ can be derived by integrating by parts thus:

$$\Gamma(-s, x) = \frac{1}{s}\left[\frac{\mathrm{e}^{-x}}{x^s} - \Gamma(1-s, x)\right]. \tag{4.30}$$

Iterating this expression then gives

$$\Gamma(1-s,x) = -\frac{(-1)^s}{(s-1)!}\left[\Gamma(0,x) + \mathrm{e}^{-x}\sum_{m=1}^{s-1}\frac{(-1)^m(m-1)!}{x^m}\right], \tag{4.31}$$

where $\Gamma(0,x) = \int_x^\infty (\mathrm{e}^{-y}/y)\,\mathrm{d}y$ is also known as the exponential integral function $-\operatorname{Ei}(-x)$ [1]. Applying this identity to (4.29) gives

$$\begin{aligned} G_0(z) &= 1 - \sum_{s=1}^{c}\binom{c}{s}\frac{1}{(s-1)!} \\ &\qquad \times \left[\mathrm{e}^{1/(1-z)}\,\Gamma\!\left(0,\frac{1}{1-z}\right) + \sum_{m=1}^{s-1}(-1)^m(m-1)!\,(1-z)^m\right] \\ &= q(z) - A_c\,\mathrm{e}^{1/(1-z)}\,\Gamma\!\left(0,\frac{1}{1-z}\right), \end{aligned} \tag{4.32}$$

where $q(z)$ is a polynomial of degree $c-1$ and $A_c = \sum_{s=1}^{c}\binom{c}{s}/(s-1)!$ depends only on $c$. For $k \geq c$, then, the degree distribution $p_k$ is given by the coefficients of $z^k$ in $-A_c\,\mathrm{e}^{1/(1-z)}\,\Gamma\big(0, 1/(1-z)\big)$. We determine these coefficients as follows. Changing the variable of integration to $x = y - z/(1-z)$, we find

$$-\mathrm{e}^{1/(1-z)}\,\Gamma\!\left(0,\frac{1}{1-z}\right) = -\mathrm{e}\int_1^\infty \frac{\mathrm{e}^{-x}}{x + z/(1-z)}\,\mathrm{d}x. \tag{4.33}$$

Then we expand the integrand to get

$$\frac{1}{x+z/(1-z)} = \frac{1}{x} - \sum_{k=1}^{\infty}\left(1-\frac{1}{x}\right)^{k-1}\frac{z^k}{x^2}. \tag{4.34}$$

Commuting the sum and the integral, we obtain

$$-\mathrm{e}^{1/(1-z)}\,\Gamma\!\left(0,\frac{1}{1-z}\right) = \sum_{k=0}^{\infty} a_k z^k, \tag{4.35}$$

where for $k = 0$,

$$a_0 = -\mathrm{e}\int_1^\infty \frac{\mathrm{e}^{-x}}{x}\,\mathrm{d}x = -\mathrm{e}\,\Gamma(0,1), \tag{4.36}$$

and for $k \geq 1$

$$a_k = \mathrm{e}\int_1^\infty \left(1-\frac{1}{x}\right)^{k-1}\frac{\mathrm{e}^{-x}}{x^2}\,\mathrm{d}x. \tag{4.37}$$

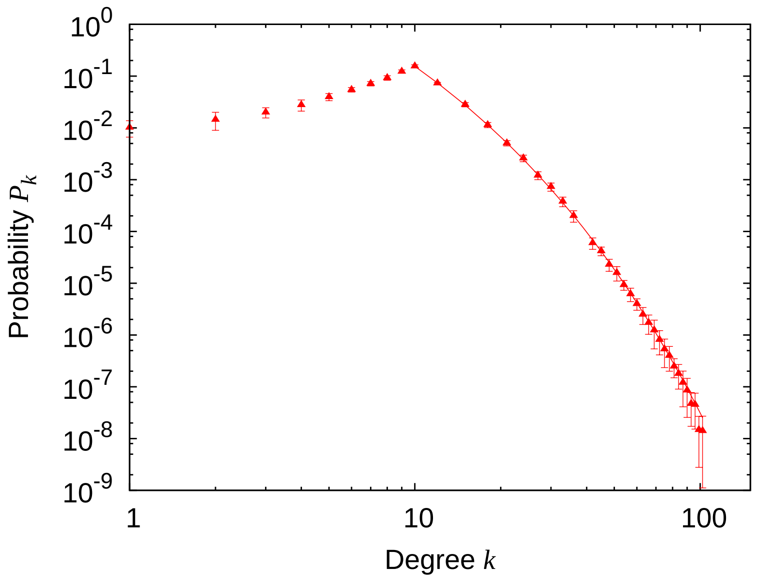

Figure 4.2: The degree distribution for our model in the case of fixed size $n = 50\,000$ and $c = 10$ with linear preferential attachment. The points represent data from our numerical simulations and the solid line is the analytic solution for $k \geq c$. Note that the tail of the distribution does not follow a power law as in growing networks with preferential attachment, but instead decays faster than a power law, as a stretched exponential.

Integrating by parts, we obtain a slightly simpler expression,

$$a_k = \frac{\mathrm{e}}{k} \int_1^{\infty} \left(1 - \frac{1}{x}\right)^k \mathrm{e}^{-x}\,\mathrm{d}x. \tag{4.38}$$

While the coefficients $a_k$ can be expressed exactly using hypergeometric functions, a perhaps more informative approach is to employ a saddle-point expansion [45]. The integrand of (4.38) is unimodal in the interval between 1 and $\infty$, and peaks at $x = \frac{1}{2}(1 + \sqrt{4k+1}) \simeq \sqrt{k}$. Approximating the integrand as a Gaussian around this point, we obtain as $k \to \infty$

$$a_k \simeq \sqrt{\pi \mathrm{e}}\, k^{-3/4}\, \mathrm{e}^{-2\sqrt{k}} \tag{4.39}$$

and $p_k = A_c\, a_k$ for $k \geq c$ as stated above.

Figure 4.2 shows the form of this solution for the case $c = 10$. Also shown in the figure are results from computer simulations of the model on systems of size $n = 50\,000$ with $c = 10$, which agree well with the analytic results. The appearance of the stretched exponential in Eq. (4.39) is worthy of note. We are aware of only a few cases of graphs with stretched exponential degree distributions that have been discussed previously, for instance in growing networks with sublinear preferential attachment [93] as well as in empirical network data [115].

### 4.3.3 Preferential attachment in a growing network

We now come to the third and most complex of our example networks, in which we combine preferential attachment with net growth of the network, $r < 1$. Logically, we should perhaps first solve the case of a growing network without preferential attachment, which in fact we have done. But the solution turns out to have no qualitatively new features to distinguish it from the constant size case and is mathematically tedious besides. Given the large amount of effort it requires and its modest rewards, therefore, we prefer to skip this case and move on to more fertile ground.

As before, perfect linear preferential attachment implies $\pi_k = k/\langle k \rangle$ or

$$\pi_k = \frac{k(1+r)}{2c}, \tag{4.40}$$

where we have made use of Eq. (4.3). Then $F_0(z) = (1+r)zG_0'(z)/2c$ and Eq. (4.9) becomes

$$G_0(z) - \frac{(1-z)}{2}\big[2r - (1+r)z\big]\,\frac{\mathrm{d}G_0}{\mathrm{d}z} = z^c. \tag{4.41}$$

An integrating factor for the left-hand side in this case is $\big|(\alpha - z)/(1-z)\big|^{-2/(1-r)}$ where $\alpha = 2r/(1+r)$. (Note that $\alpha < 1$ when $r < 1$.) Unfortunately, this integrating factor is non-analytic at $z = \alpha$, which makes integrals traversing this point cumbersome. To circumvent this difficulty, we observe that the second term in Eq. (4.41) vanishes at $z = \alpha$, giving $G_0(\alpha) = \alpha^c$. This provides us with an alternative boundary condition on $G_0(z)$, allowing us to fix the integrating constant while only integrating up to $z =$

$\alpha$. It is then straightforward to show that

$$\begin{aligned} G_0(z) &= \frac{2}{1+r}\left(\frac{\alpha-z}{1-z}\right)^{-2/(1-r)} \\ &\quad \times \int_z^{\alpha}\left(\frac{\alpha-t}{1-t}\right)^{2/(1-r)}\frac{t^c\,\mathrm{d}t}{(1-t)(\alpha-t)}, \end{aligned} \tag{4.42}$$

for $z \le \alpha$. Since the degree distribution is entirely determined by the behavior of $G_0(z)$ at the origin, it is adequate to restrict our solution to this regime.

Changing variables to $u = (\alpha - t)/(1-\alpha)$, we find

$$\begin{aligned} G_0(z) &= \frac{2}{1+r}\left(\frac{\alpha-z}{1-z}\right)^{1-\gamma}(1-\alpha)^{-1} \\ &\quad \times \int_0^{\frac{\alpha-z}{1-\alpha}}\left(\frac{u}{1+u}\right)^{\gamma}\big[\alpha-(1-\alpha)u\big]^c\,\frac{\mathrm{d}u}{u^2}, \end{aligned} \tag{4.43}$$

where $\gamma = (3-r)/(1-r)$. If we expand the last factor in the integrand, this becomes

$$\begin{aligned} G_0(z) &= \frac{2}{1+r}\sum_{s=0}^{c}(-1)^s\binom{c}{s}(1-\alpha)^{s-1}\alpha^{c-s} \\ &\quad \times \left(\frac{\alpha-z}{1-z}\right)^{1-\gamma}\int_0^{\frac{\alpha-z}{1-\alpha}}\frac{u^{s+\gamma-2}}{(1+u)^{\gamma}}\,\mathrm{d}u. \end{aligned} \tag{4.44}$$

We observe the following useful identity:

$$\begin{aligned} \int_0^x \frac{u^{\beta}}{(1+u)^{\gamma}}\,\mathrm{d}u &= \int_0^x\left(\frac{u}{1+u}\right)^{\beta}(1+u)^{\beta-\gamma}\,\mathrm{d}u \\ &= \frac{x^{\beta}}{(\beta-\gamma+1)(1+x)^{\gamma-1}} - \frac{\beta}{\beta-\gamma+1}\int_0^x\frac{u^{\beta-1}}{(1+u)^{\gamma}}\,\mathrm{d}u, \end{aligned} \tag{4.45}$$

where the second equality is derived via integration by parts. Setting $\beta = s+\gamma-2$ and $x = (\alpha-z)/(1-\alpha)$ and noting that the last integral has the same form as the first, we can employ this identity iteratively $s-1$ times to get

$$\begin{aligned} \left(\frac{\alpha-z}{1-z}\right)^{1-\gamma}\int_0^{\frac{\alpha-z}{1-\alpha}}\frac{u^{s+\gamma-2}}{(1+u)^{\gamma}}\,\mathrm{d}u &= (-1)^{s+1}\frac{\Gamma(s+\gamma-1)}{\Gamma(s)} \\ &\quad \times \left[\frac{1}{\Gamma(\gamma)}\left(\frac{\alpha-z}{1-z}\right)^{1-\gamma}\int_0^{\frac{\alpha-z}{1-\alpha}}\frac{u^{\gamma-1}}{(1+u)^{\gamma}}\,\mathrm{d}u \right. \\ &\qquad \left. + \sum_{m=1}^{s-1}\frac{(-1)^m}{\Gamma(\gamma+m)}\left(\frac{\alpha-z}{1-z}\right)^{m}\right]. \end{aligned} \tag{4.46}$$

The final sum can be evaluated in closed form in terms of the incomplete Γ-function, but our primary focus here is on the preceding term. Substituting into Eq. (4.44), we see that $G_0(z) = q(z) + A_{c,r}h(z)$, where

$$h(z) = -\left(\frac{\alpha - z}{1 - z}\right)^{1-\gamma} \int_0^{\frac{\alpha-z}{1-\alpha}} \frac{u^{\gamma-1}}{(1+u)^{\gamma}}\, \mathrm{d}u, \tag{4.47}$$

$$A_{c,r} = \frac{2}{1+r} \sum_{s=0}^{c} \binom{c}{s} (1-\alpha)^{s-1} \alpha^{c-s} \frac{\Gamma(\gamma + s - 1)}{\Gamma(\gamma)\Gamma(s)}, \tag{4.48}$$

and $q(z)$ is a polynomial of order $c - 1$ in $z$.

Since $A_{c,r}$ depends only on $c$ and $r$ and $q(z)$ has no terms in $z$ of order $z^c$ or higher, the degree distribution for $k \geq c$ is, to within a multiplicative constant, given by the coefficients in the expansion of $h(z)$ about zero. Making the change of variables

$$u = \frac{y}{(1-z)/(\alpha - z) - y}, \tag{4.49}$$

we find that

$$h(z) = -\int_0^1 \frac{y^{\gamma-1}\, \mathrm{d}y}{(1-z)/(\alpha-z) - y}, \tag{4.50}$$

and expanding the integrand in powers of $z$ we obtain $h(z) = \sum_{k=0}^{\infty} a_k z^k$ with

$$\begin{aligned} a_k &= (1-\alpha) \int_0^1 \frac{(1-y)^{k-1}}{(1-\alpha y)^{k+1}}\, y^{\gamma-1}\, \mathrm{d}y \\ &= \frac{\gamma-1}{k} \int_0^1 \left(\frac{1-y}{1-\alpha y}\right)^k y^{\gamma-2}\, \mathrm{d}y, \end{aligned} \tag{4.51}$$

for $k \geq 1$, where the second equality follows via an integration by parts.

As in the case of constant size, we can express these coefficients in closed form using special functions, but if we are primarily interested in the form of the tail of the degree distribution then a more revealing approach is to make a further substitution $y = x/k$, giving

$$a_k = (\gamma - 1)\, k^{-\gamma} \int_0^k \frac{(1 - x/k)^k}{(1 - \alpha x/k)^k}\, x^{\gamma-2}\, \mathrm{d}x. \tag{4.52}$$

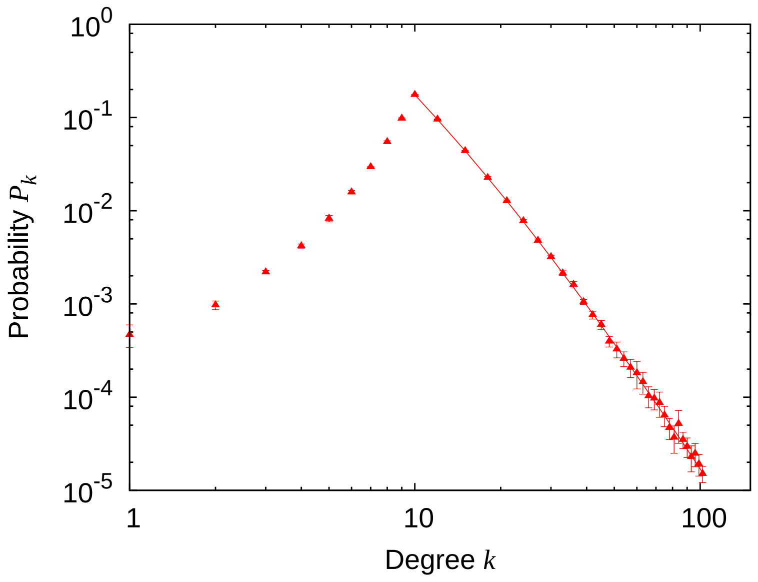

Figure 4.3: Degree distribution for a growing network with linear preferential attachment and $r = \frac{1}{2}$, $c = 10$. The solid line represents the analytic solution, Eqs. (4.48), and (4.51), for $k \geq c$ and the points represent simulation results for systems with final size $n = 100\,000$ vertices.

In the limit of large $k$ this becomes

$$\begin{aligned} a_k &\simeq (\gamma - 1)\, k^{-\gamma} \int_0^\infty \mathrm{e}^{-(1-\alpha)x} x^{\gamma-2}\, \mathrm{d}x \\ &= \frac{\Gamma(\gamma)}{(1-\alpha)^{\gamma-1}}\, k^{-\gamma}, \end{aligned} \tag{4.53}$$

and $p_k = A_{c,r}\, a_k$ for $k \geq c$ as stated above.

Thus the tail of the degree distribution follows a power law with exponent

$$\gamma = \frac{3-r}{1-r}. \tag{4.54}$$

Note that this exponent diverges as $r \to 1$ so that the power law becomes ever steeper as the growth rate slows, eventually assuming the stretched exponential form of Eq. (4.39)—steeper than any power law—in the limit $r = 1$. In the limit $r \to 0$ we recover the established power-law behavior $a_k \sim k^{-3}$ for growing graphs with preferential attachment and no vertex removal [49, 13, 54, 96].

In Fig. 4.3 we show the form of the degree distribution for this model for the case $r = \frac{1}{2}$, $c = 10$, along with numerical results from simulations of the model on networks of (final) size $n = 100\,000$ vertices. The power-law behavior is clearly visible on the logarithmic scales used as a straight line in the tail of the distribution. Once again the analytic solution and simulations are in excellent agreement.

We note that Sarshar and Roychowdhury [136] and, subsequently, Chung and Lu [40] and Cooper, Frieze, and Vera [43], independently demonstrated power-law behavior in the degree distribution of networks in the case $r < 1$. Their results focus on the tail of the distribution rather than on exact solutions, but they find the same dependence of the exponent on the growth rate.

## 4.4 Discussion

In this chapter we have studied models of the time evolution of networks in which, in addition to the widely considered case of addition of vertices, we also include vertex removal. We have given exact solutions for cases in which vertices are added and removed at the same rate, a potential model for steady-state networks such as peer-to-peer networks, and cases in which the rate of addition exceeds the rate of removal, which we regard as a simple model for the growth of, for example, the worldwide web.

We find very different behaviors in these various cases. For a steady-state network in which newly added vertices attach to others at random we find a degree distribution, Eqs. (4.18) and (4.19), which is sharply peaked about its maximum and has a rapidly decaying (Poisson) tail. This distribution is quite unlike the right-skewed degree distributions found in many real-world networks, but as a possible form for a “designed” network such as a peer-to-peer network it might be preferable over skewed forms, being more homogeneous and hence distributing traffic more evenly.

If newly appearing vertices attach to others using a linear preferential attachment mechanism, whereby vertices gain new edges in proportion to the number they already possess, we find that the degree distribution becomes a stretched exponential, Eqs. (4.38) and (4.39), a substantially broader distribution than that of the random attachment case, though still more rapidly decaying than the power laws often seen in growing networks.

And in the case where the network shows net growth, adding vertices faster than it loses them, we find that the degree distribution follows a power law, Eqs. (4.51) and (4.53), with an exponent $\gamma$ that assumes values in the range $3 \leq \gamma < \infty$, diverging as the growth rate tends to zero.

This last result is of interest for a number of reasons. First, it shows that power-law behavior can be rigorously established in networks that grow but also lose vertices. Most previous analytic models of network growth have focused solely on vertex addition. And while the real worldwide web and other networks appear to have degree distributions that closely follow power laws, these networks also clearly lose vertices as well as gaining them. The results presented here demonstrate that the widely studied mechanism of preferential attachment for generating power-law behavior also works in this regime.

On the other hand, the large values of the exponent $\gamma$ generated by our model appear not to be in agreement with the behavior observed in real-world networks, most of which have exponents in the range from 2 to 3 [6, 52, 108]. There are well-known mechanisms that can reduce the exponent from 3 to values slightly lower—specifically the generalization of the preferential attachment model to the case of a directed network [49, 54], which is in any case a more appropriate model for the worldwide web. In the limit of low growth rate, however, our model predicts a diverging exponent and, while the exact value may not be accurate because of a host of complicating factors, it seems likely that the divergence itself is a robust phenomenon; as other authors have commented, there are good reasons to believe that net growth is one of the fundamental requirements for the generation of power-law degree distributions by the kind of mechanisms considered here.

Thus the fact that we do not observe very large exponents in real networks appears to indicate that most networks are in a regime where growth dominates over vertex loss by a wide margin. It is possible however that this will not always be the case. The web, for example, has certainly being enjoying a period of very vigorous growth since its appearance in the early 1990s, but it could be that this is a sign primarily of its youth, and that as the network matures its size will grow more slowly, the vertices added being more nearly balanced by those taken away. Were this to happen, we would expect to see the exponent of the degree distribution grow larger. A sufficiently large exponent would make the distribution indistinguishable experimentally from an exponential or stretched exponential distribution, although we do not realistically anticipate seeing behavior of this type any time in the near future.

# CHAPTER V

# Network growth models II: Design and generation of networks with desired properties

## 5.1 Introduction

In the previous chapter, we proposed and examined a model for a network that evolved via the addition/deletion of nodes and edges using a variety of schemes. Here we will use the insight that the model afforded us—particularly our first example solution in Sec. 4.3.1—to use it for a practical application.

Driven by the fact that degrees of vertices have a strong effect on the overall behavior of a network, a large amount of effort has been devoted to the study of degree distributions, their measurement and the formulations of theories to explain how they take their observed forms, and models of the effect of particular degree distributions on dynamical processes on networks, network resilience, percolation and many other phenomena—indeed much of the material of the preceding chapters has dealt with precisely these matters. Such studies are appropriate for "naturally occurring" networks, whose structure evolves under rules not directly under our control. Representative examples are the Internet, World Wide Web and a majority of social networks, which though man-made are distributed in nature, in the sense that their evolution is not governed by a central authority.

There is a different class of networks, mostly infrastructure related, such as the transportation and power grids, communication networks such as telephone networks, that are designed by a centrally controlled authority. For these networks, it is worthwhile to determine if one can design a structure that optimizes some desired property. For instance, Paul *et al.* [124] have considered how the structure of a network should be chosen to optimize the network's robustness to deletion of its vertices.

Finally, there are a relatively new class of networks that falls in between these two types, the best known example being peer-to-peer file-sharing networks. These networks grow in a collaborative, distributed fashion so that although we have no direct influence over their structure, we can manipulate some of the rules by which they form, giving us a limited but potentially useful influence over their properties. A peer-to-peer filesharing network is a virtual network of linked computers that share data among themselves. The network is formed by a dynamical process under which individual computers continually join or leave the network, and the rules of joining and leaving can be manipulated to some extent by changing the behavior of the software governing computers' behaviors. It is well established that the structure of peer-to-peer networks can have a strong effect on their performance [3, 135] but to a large extent that structure has in the past been regarded as an experimentally determined quantity [79]. Here we consider ways in which the structure can be manipulated by changing the behavior of individual nodes so as to optimize network performance.

## 5.2 Growing networks with desired properties

In this chapter we focus primarily on creating networks with desired degree distributions. There are two basic problems we need to address if we want to create a network with a specific degree distribution solely by manipulating the rules by which vertices enter and leave the network. First, we need to find rules that will achieve the desired result, and second, we need to find a practical mechanism that implements those rules and operates in reasonable time. We deal with these questions in order.

Our approach to growing a network with a desired degree distribution is based on the idea of the attachment kernel $\pi_k$ discussed in Sec. 4.2. We assume that vertices join our network at intervals and that when they do so they form connections—edges—to some number of other vertices in the network. By designing the software appropriately, we can in a peer-to-peer network choose the number of edges a newly joining vertex makes and also, as we will shortly show, some crucial aspects of which other vertices those edges connect to. It is the attachment kernel that we will manipulate to produce a desired degree distribution.

In a peer-to-peer network users may exit the network whenever they want and we as designers have little control over this aspect of the network dynamics. We will assume in the calculations that follow that vertices simply vanish at random. We will also assume that, on the typical time-scales over which people enter and leave the network, the total size $n$ of the network does not change substantially, so that the rates at which vertices enter and leave are roughly equal. For simplicity let us say that exactly one vertex enters the network and one leaves per unit time (although the results presented here are in fact still valid even if only the probabilities per unit time of addition and deletion of vertices are equal and not the rates).

Now let us chose the initial degrees of vertices when they join the network, i.e., the number of connections that they form upon entering, at random from some distribution $r_k$. Building on our previous results in Sec. 4.2.1, we observe that the evolution of the degree distribution of our network can be described by a rate equation as follows. The number of vertices with degree $k$ at a particular time is $np_k$. One unit of time later we have added one vertex and taken away one vertex, so that the number with degree $k$ becomes

$$np'_k = np_k + c\pi_{k-1}p_{k-1} - c\pi_k p_k + (k+1)p_{k+1} - kp_k - p_k + r_k, \tag{5.1}$$

with the convention that $p_{-1} = 0$, and $c = \sum_{k=0}^{\infty} kr_k$, which is the average degree of vertices added to the network. The terms $c\pi_{k-1}p_{k-1}$ and $-c\pi_k p_k$ in Eq. (5.1) represent the flow of vertices with degree $k-1$ to $k$ and $k$ to $k+1$, as they gain extra edges with the addition of new vertices. The terms $(k+1)p_{k+1}$ and $-kp_k$ represent the flow of vertices with degree $k+1$ and $k$ to $k$ and $k-1$, as they lose edges with the removal of neighboring vertices. The term $-p_k$ represents the probability of removal of a vertex of degree $k$ and the term $r_k$ represents the addition of a new vertex with degree $k$ to the network.

As before, we assume $p_k$ has an asymptotic form in the limit of large time, and set $p'_k = p_k$:

$$c\pi_{k-1}p_{k-1} - c\pi_k p_k + (k+1)p_{k+1} - kp_k - p_k + r_k = 0. \tag{5.2}$$

Defining the standard generating functions for the degree distribution as well as for the degrees of vertices added and for the attachment kernel,

$$G_0(z) = \sum_{k=0}^{\infty} p_k z^k, \tag{5.3}$$

$$H_0(z) = \sum_{k=0}^{\infty} r_k z^k, \tag{5.4}$$

$$F_0(z) = \sum_{k=0}^{\infty} \pi_k p_k z^k, \tag{5.5}$$

and multiplying both sides of (5.2) by $z^k$ and summing over $k$, we then find that the generating functions satisfy the differential equation

$$(1-z)\frac{\mathrm{d}G_0}{\mathrm{d}z} - G_0(z) - c(1-z)F_0(z) + H_0(z) = 0. \tag{5.6}$$

We are interested in creating a network with a given degree distribution, i.e., with a given $G_0(z)$. Rearranging (5.6), we find that the choice of attachment kernel $\pi_k$ that achieves this is such that

$$F_0(z) = \frac{1}{c}\left[\frac{\mathrm{d}G_0}{\mathrm{d}z} + \frac{H_0(z) - G_0(z)}{1-z}\right]. \tag{5.7}$$

Taking the limit $z \to 1$, noting that normalization requires that all the generating functions tend to 1 at $z = 1$, and applying L'Hopital's rule, we find

$$1 = \frac{1}{c}\left[\langle k\rangle + \langle k\rangle - c\right], \tag{5.8}$$

where we have made use of the fact that the average degree in the network is $\langle k\rangle = G_0'(1)$ and $c = H_0'(1)$. Rearranging, we then find that $c = \langle k\rangle$. In other words, solutions to Eq. (5.6) require that the average degree $c$ of vertices added to the network be equal to the average degree of vertices in the network as a whole. Making use of this result, we can write Eq. (5.7) in the form

$$F_0(z) = G_1(z) + \frac{H_0(z) - G_0(z)}{c(1-z)}, \tag{5.9}$$

where $G_1(z) = G_0'(z)/G_0'(1) = \sum_k q_k z^k$ is the generating function for the excess degree distribution

$$q_k = \frac{(k+1)p_{k+1}}{\langle k\rangle}, \tag{5.10}$$

Now it is straightforward to derive the desired attachment kernel. Noting that

$$\frac{1}{1-z} = \sum_{k=0}^{\infty} z^k, \tag{5.11}$$

we can simply read off the coefficient of $z^k$ on either side of Eq. (5.9), to give

$$\pi_k p_k = q_k + \frac{1}{c}\sum_{m=0}^{k}(r_m - p_m), \tag{5.12}$$

or equivalently

$$\pi_k = \frac{1}{cp_k}\big[(k+1)p_{k+1} + P_{k+1} - R_{k+1}\big], \tag{5.13}$$

where $P_k$ is the cumulative distribution of vertex degrees and $R_k$ is the cumulative distribution of added degrees:

$$P_k = \sum_{m=k}^{\infty} p_m, \qquad R_k = \sum_{m=k}^{\infty} r_m. \tag{5.14}$$

Since we are at liberty to choose both $r_k$ and $\pi_k$, we have many options for satisfying Eq. (5.13); given (almost) any choice of the distribution $r_k$ of the degrees of added vertices, we can find the corresponding $\pi_k$ that will give the desired final degree distribution of the network. One simple choice would be to make the degree distribution of the added vertices the same as the desired degree distribution, so that $R_k = P_k$. Then

$$\pi_k = \frac{q_k}{p_k} = \frac{(k+1)p_{k+1}}{cp_k}. \tag{5.15}$$

In other words, if we have some desired degree distribution $p_k$ for our network, one way to achieve it is to add vertices with exactly that degree distribution and then arrange the attachment process so that the degree distribution remains preserved thereafter, even as vertices and edges are added to and removed from the network. Equation (5.15) tells us the choice of attachment kernel that will achieve this. Equation (5.15) will work for essentially any choice of degree distribution $p_k$, except choices for which $p_k = 0$ and $p_{k+1} > 0$ for some $k$. In the latter case Eq. (5.15) will diverge for some value(s) of $k$.

### 5.2.1 Example: power-law degree distribution

As an example, consider the creation of a network with a power-law degree distribution. Adamic *et al.* [3] have shown that search processes on peer-to-peer networks with power-law degree distributions are particularly efficient, so there are reasons why one might want to generate such a network.

Let us choose

$$p_k = \begin{cases} Ck^{-\gamma} & \text{for } k \geq 1, \\ p_0 & \text{for } k = 0, \end{cases} \tag{5.16}$$

where $\gamma$ and $p_0$ are constants and the normalizing factor $C$ is given by

$$C = \frac{1 - p_0}{\zeta(\gamma)}, \tag{5.17}$$

where $\zeta(\gamma)$ is the Riemann zeta-function [1]. Then the mean degree is

$$\langle k \rangle = c = (1 - p_0)\frac{\zeta(\gamma - 1)}{\zeta(\gamma)}, \tag{5.18}$$

and Eq. (5.15) tells us that the correct choice of attachment kernel in this case is

$$\pi_k = \frac{1}{1 - p_0} \frac{\zeta(\gamma)}{\zeta(\gamma - 1)} \frac{k^\gamma}{(k+1)^{\gamma - 1}}, \tag{5.19}$$

for $k \geq 1$ and

$$\pi_0 = \frac{1}{p_0 \zeta(\gamma - 1)}. \tag{5.20}$$

It is interesting to note that as $k$ becomes large, this attachment kernel goes as $\pi_k \sim k$, the so-called (linear) preferential attachment form in which vertices connect to others in simple proportion to their current degree. In growing networks this form is known to give rise, asymptotically, to a power-law degree distribution—see Chap. IV. It is important to understand, however, that in the present case the network is not growing and hence, despite the apparent similarity, this is not the same result. Indeed, it is known that for non-growing networks, purely linear preferential attachment does not produce power-law degree distributions [136, 43], but instead generates stretched exponential distributions [103]. Thus it is somewhat surprising to observe that one can, nonetheless, create a power-law degree distribution in a non-growing network using an attachment kernel that seems, superficially, quite close to the linear form.

Sarshar and Roychowdhury [136] showed previously that it is possible to generate a non-growing power-law network by using linear preferential attachment and then compensating for the expected loss of power-law behavior by rewiring the connections of some vertices after their addition to the network. Our results indicate that, although this process will certainly work, it is not necessary: a slight modification to the preferential attachment process will achieve the same goal and frees us from the need to rewire any edges.

Note also that (5.19) is not the only solution of Eq. (5.13) that will generate a power-law distribution. If we choose a different (e.g., non-power-law) distribution for the vertices added to the network, we can still generate an overall power-law distribution by choosing the attachment kernel to satisfy Eq. (5.13). Suppose, for instance, that, rather than adding vertices with a power-law degree distribution, we prefer to give them a Poisson distribution with mean $c$:

$$r_k = \mathrm{e}^{-c}\,\frac{c^k}{k!}. \tag{5.21}$$

In this case $R_k = 1 - \Gamma(k,c)/\Gamma(k)$, where $\Gamma(k)$ is the standard gamma function and $\Gamma(k,c)$ is the incomplete gamma function. Then the power law is correctly generated by the choice

$$\begin{aligned}\pi_k = {} & \frac{1}{1-p_0}\,\frac{\zeta(\gamma)}{\zeta(\gamma-1)}\,k^{\gamma} \\ & \times\left[(k+1)^{-\gamma+1} + \zeta(\gamma,k+1) - \frac{\zeta(\gamma)}{1-p_0}\left(1 - \frac{\Gamma(k+1,c)}{\Gamma(k+1)}\right)\right],\end{aligned} \tag{5.22}$$

for $k \geq 1$, where $\zeta(\gamma, x)$ is the generalized zeta function $\zeta(\gamma, x) = \sum_{k=0}^{\infty}(k+x)^{-\gamma}$ and for $k = 0$,

$$\pi_0 = \frac{1}{p_0 \zeta(\gamma - 1)} \left[ 1 + \frac{\mathrm{e}^{-c} - p_0}{1 - p_0} \zeta(\gamma) \right]. \tag{5.23}$$

## 5.3 A practical implementation

In theory, we should be able use the ideas of the previous section to grow a network with a desired degree distribution. This does not, however, yet mean we can do so in practice. To make our scheme a practical reality, we still need to devise a realistic way to place edges between vertices with the desired attachment kernel $\pi_k$. If each vertex entering the network knew the identities and degrees of all other vertices, this would be easy: we would simply select a degree $k$ at random in proportion to $\pi_k p_k$, and then attach our new edge to a vertex chosen uniformly at random from those having that degree.

In the real world, however, and particularly in peer-to-peer networks, no vertex "knows" the identity of all others. Typically, computers only know the identities (such as IP addresses) of their immediate network neighbors. To get around this problem, we propose the following scheme, which makes use of biased random walks.

A random walk, in this context, is a succession of steps along edges in our network where at each vertex $i$ we choose to step next to a vertex chosen at random from the set of neighbors of $i$. In the context of a peer-to-peer computer network, for example, such a walk can be implemented by message passing between peers. The "walker" is a message or data packet that is passed from computer to neighboring computer, with each computer making random choices about which neighbor to pass to next.

Starting a walk from any vertex in the network, we can sample vertices by allowing the walk to take some fixed number of steps and then choosing the vertex that it lands upon on its final step. We will consider random walks in which the choice of which step to make at each vertex is deliberately biased to create a desired probability distribution for the sample as follows. The use of random walks in this way has been considered previously by Gkantsidis et al. [67], who found that in certain circumstances random walks can provide an efficient mechanism for growing appropriately structured networks. They however, considered only unbiased walks, which limits the range of possible outcomes. In our work, by contrast, we consider walks in which the choice of which step to make at each vertex is deliberately biased to create a desired probability distribution for the sample as follows.

Consider a walk in which a walker at vertex $j$ chooses uniformly at random one of the $k_j$ neighbors of that vertex. Let us call this neighbor $i$. Then the walk takes a step to vertex $i$ with some acceptance probability $P_{ij}$. The total probability $T_{ij}$ of a transition from $j$ to $i$ given that we are currently at $j$ is

$$T_{ij} = \frac{A_{ij}}{k_j} P_{ij}, \tag{5.24}$$

where $k_j$ is the degree of vertex $j$ and $A_{ij}$ is an element of the adjacency matrix:

$$A_{ij} = \begin{cases} 1 & \text{if there is an edge joining vertices } i, j, \\ 0 & \text{otherwise.} \end{cases} \tag{5.25}$$

If the step is not accepted, then the random walker remains at vertex $j$ for the current step.

This random walk constitutes an ordinary Markov process, which converges to a distribution $p_i$ over vertices provided the network is connected (i.e., consists of a single component) and provided $T_{ij}$ satisfies the detailed balance condition

$$T_{ij} p_j = T_{ji} p_i. \tag{5.26}$$

In the present case we wish to select vertices in proportion to the attachment kernel $\pi_k$. Setting $p_i = \pi_{k_i}$, this implies that $T_{ij}$ should satisfy

$$\frac{T_{ij}}{T_{ji}} = \frac{p_i}{p_j} = \frac{\pi_{k_i}}{\pi_{k_j}}. \tag{5.27}$$

Or, making use of Eqs. (5.15) and (5.24) for the case where $r_k = p_k$, we find

$$\frac{P_{ij}}{P_{ji}} = \frac{(k_i+1)p_{k_i+1}}{k_i p_{k_i}} \frac{k_j p_{k_j}}{(k_j+1)p_{k_j+1}} = \frac{q_{k_i} q_{k_j-1}}{q_{k_j} q_{k_i-1}}, \tag{5.28}$$

where $q_k$ is again the excess degree distribution, Eq. (5.10).

In practice, we can satisfy this equation by making the standard Metropolis-Hastings choice [114] for the acceptance probability:

$$P_{ij} = \begin{cases} q_{k_i} q_{k_j-1}/q_{k_j} q_{k_i-1} & \text{if } q_{k_i}/q_{k_i-1} < q_{k_j}/q_{k_j-1}, \\ 1 & \text{otherwise.} \end{cases} \tag{5.29}$$

Thus the calculation of the acceptance probability requires only that each vertex know the degrees of its neighboring vertices, which can be established by a brief exchange of data when the need arises.

As an example, suppose we wish to generate a network with a Poisson degree distribution

$$p_k = \mathrm{e}^{-\mu} \frac{\mu^k}{k!}, \tag{5.30}$$

where $\mu$ is the mean of the Poisson distribution. Then we find that the appropriate choice of acceptance ratio is

$$P_{ij} = \begin{cases} k_j/k_i & \text{if } k_i > k_j, \\ 1 & \text{otherwise.} \end{cases} \tag{5.31}$$

(As discussed above, we must also make sure to choose the mean degree $c$ of vertices added to the network to be equal to $\mu$.)

Our proposed method for creating a network is thus as follows. Each newly joining vertex $i$ first chooses a degree $k$ for itself, which is drawn from the desired distribution $p_k$. It must also locate one single other vertex $j$ in the network. It might do this for instance using a list of known previous members of the network or a standardized list of permanent network members. Vertex $j$ is probably not selected randomly from the network, so it is *not* chosen as a neighbor of $i$. Instead, we use it as the starting point for a set of $k$ biased random walkers of the type described above. Each walker consists of a message, which starts at $j$ and propagates through the network by being passed from computer to neighboring computer. The message contains (at a minimum) the address of the computer at vertex $i$ as well as a counter that is updated by each computer to record the number of steps the walker has taken. (Bear in mind that steps on which the walker doesn't move, because the proposed move was rejected, are still counted as steps.) The computer that the walker reaches on its $t$'th step, where $t$ is a fixed but generous constant chosen to allow enough time for mixing of the walk, establishes a new network edge between itself and vertex $i$ and the walker is then deleted. When all $k$ walkers have terminated in this way, vertex $i$ has $k$ new neighbors in the network, chosen in proportion to the correct attachment kernel $\pi_k$ for the desired distribution. After a suitable interval of time, this process will result in a network that has the chosen degree distribution $p_k$, but is otherwise random.

As a test of this method, we have performed simulations of the growth of a network with a Poisson degree distribution as in Eq. (5.31). Starting from a random graph of the desired size $n$, we randomly add and remove vertices according to the prescription given above. Figure 5.1 shows the resulting degree distribution for the case $\mu = 10$, along with the expected Poisson distribution. As the figure shows, the agreement

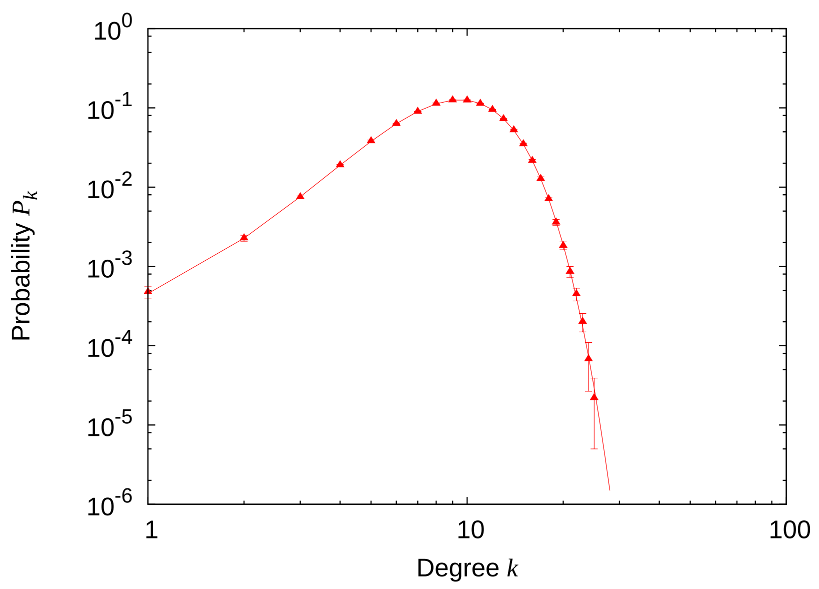

Figure 5.1: The degree distribution for a network of $n = 50\,000$ vertices generated using the biased random walk mechanism described in the text with $\mu = 10$. The points represent the results of our simulations and the solid line is the target distribution, Eq. (5.30).

between the two is excellent.

## 5.4 Example application

As an example of the application of these ideas we consider peer-to-peer networks. Bandwidth restrictions and search times place substantial constraints on the performance of peer-to-peer networks, and the methods of the previous sections can be used to nudge networks towards a structure that improves their performance in these respects. More sophisticated applications are certainly possible, but the one presented here offers an indication of the kinds of possibilities open to us.

### 5.4.1 Definition of the problem

Consider a distributed database consisting of a set of computers each of which holds some data items. Copies of the same item can exist on more than one computer, which would make searching easier, but we will not assume this to be the case. Computers are connected together in a "virtual network," meaning that each computer is designated as a "neighbor" of some number of other computers. These connections between computers are purely notional: every computer can communicate with every other directly over the Internet or other physical network. The virtual network is used only to limit the amount of information that computers have to keep about their peers.

Each computer maintains a directory of the data items held by its network neighbors, but not by any other computers in the network. Searches for items are performed by passing a request for a particular item from computer to computer until it reaches one in whose directory that item appears, meaning that one of that computer's neighbors holds the item. The identity of the computer holding the item is then transmitted back to the origin of the search and the origin and target computers communicate directly thereafter to negotiate the transfer of the item. This basic model is essentially the same as that used by other authors [3] as well as by many actual peer-to-peer networks in the real world. Note that it achieves efficiency by the use of relatively large directories at each vertex of the network, which inevitably use up memory resources on the computers. However, with standard hash-coding techniques and for databases of the typical sizes encountered in practical situations (thousands or millions of items) the amounts of memory involved are quite modest by modern standards.

### 5.4.2 Search time and bandwidth

The two metrics of search performance that we consider in this example are *search time* and *bandwidth*, both of which should be low for a good search algorithm. We define the search time to be the number of steps taken by a propagating search query before the desired target item is found. We define the bandwidth for a vertex as the average number of queries that pass through that vertex per unit time. Bandwidth is a measure of the actual communications bandwidth that vertices must expend to keep the network as a whole running smoothly, but it is also a rough measure of the CPU time they must devote to searches. Since these are limited resources it is crucial that we not allow the bandwidth to grow too quickly as vertices are added to the network, otherwise the size of the network will be constrained, a severe disadvantage for networks that can in some cases swell to encompass a significant fraction of all the computers on the planet. (In some early peer-to-peer networks, issues such as this did indeed place impractical limits on network size [127, 128].)

Assuming that the average behavior of a user of the database remains essentially the same as the network gets larger, the number of queries launched per unit time should increase linearly with the size of the network, which in turn suggests that the bandwidth per vertex might also increase with network size, which would be a bad thing. As we will show, however, it is possible to avoid this by designing the topology of the network appropriately.

### 5.4.3 Search strategies and search time

In order to treat the search problem quantitatively, we need to define a search strategy or algorithm. Here we consider a very simple—even brainless—strategy, again employing the idea of a random walk. This *random walk search* is certainly not the most efficient strategy possible, but it has two significant advantages for our purposes. First, it is simple enough to allow us to carry out analytic calculations of its performance. Second, as we will show, even this basic strategy can be made to work very well. Our results constitute an existence proof that good performance is achievable: searches are necessarily possible that are at least as good as those analyzed here.

The definition of our random walk search is simple: the vertex $i$ originating a search sends a query for the item it wishes to find to one of its neighbors $j$, chosen uniformly at random. If that item exists in the neighbor's directory the identity of the computer holding the item is transmitted to the originating vertex and the search ends. If not, then $j$ passes the query to one of its neighbors chosen at random, and so forth. (One obvious improvement to the algorithm already suggests itself: that $j$ not pass the query back to $i$ again. As we have said, however, our goal is simplicity and we will allow such "backtracking" in the interests of simplifying the analysis.)

We can study the behavior of this random walk search by a method similar to the one we employed for the analysis of the biased random walks of Section 5.3. Let $p_i$ be the probability that our random walker is at vertex $i$ at a particular time. Then the probability $p'_i$ of its being at $i$ one step later, assuming the target item has not been found, is

$$p'_i = \sum_j \frac{A_{ij}}{k_j} p_j, \tag{5.32}$$

where $k_j$ is the degree of vertex $j$ and $A_{ij}$ is an element of the adjacency matrix, Eq. (5.25). Under the same conditions as before the probability distribution over vertices then tends to the fixed point of (5.32), which is at

$$p_i = \frac{k_i}{2m}, \tag{5.33}$$

where $m$ is the total number of edges in the network. That is, the random walk visits vertices with probability proportional to their degrees. (An alternative statement of the same result is that the random walk visits *edges* uniformly.)

When our random walker arrives at a previously unvisited vertex of degree $k_i$, it "learns" from that vertex's directory about the items held by all immediate neighbors of the vertex, of which there are $k_i - 1$ excluding the vertex we arrived from (whose items by definition we already know about). Thus at every step the walker gathers more information about the network. The average number of vertices it learns about upon making a single step is $\sum_i p_i(k_i - 1)$, with $p_i$ given by (5.33), and hence the total number it learns about after $\tau$ steps is

$$\frac{\tau}{2m} \sum_i k_i(k_i - 1) = \tau \left[ \frac{\langle k^2 \rangle}{\langle k \rangle} - 1 \right], \tag{5.34}$$

where $\langle k \rangle$ and $\langle k^2 \rangle$ represent the mean and mean-square degrees in the network and we have made use of $2m = n\langle k \rangle$. (There is in theory a correction to this result because the random walker is allowed to backtrack and visit vertices visited previously. For a well-mixed walk, however, this correction is of order $1/\langle k \rangle$, which, as we will see, is negligible for the networks we will be considering.)

How long will it take our walker to find the desired target item? That depends on how many instances of the target exist in the network. In many cases of practical interest, copies of items exist on a fixed *fraction* of the vertices in the network, which makes for quite an easy search. We will not however assume this to be the case here. Instead we will consider the much harder problem in which copies of the target item exist on only a fixed *number* of vertices, where that number could potentially be just 1. In this case, the walker will need to learn about the contents of $\mathrm{O}(n)$ vertices in order to find the target and hence the average time to find the target is given by

$$\tau \left[ \frac{\langle k^2 \rangle}{\langle k \rangle} - 1 \right] = An, \tag{5.35}$$

for some constant $A$, or equivalently,

$$\tau = A \frac{n}{\langle k^2 \rangle / \langle k \rangle - 1}. \tag{5.36}$$

This equation is related to previous results by Broder *et al.*. [29], who showed that the number of steps required for a random walker to visit a fixed fraction of vertices in a network can be expressed in terms of the second eigenvalue of the transition matrix for the walk.

Consider, for instance, a network with a power-law degree distribution of the form, $p_k = Ck^{-\gamma}$, where $\gamma$ is a positive exponent and $C$ is a normalizing constant chosen such that $\sum_{k=0}^{\infty} p_k = 1$. Real-world networks usually exhibit power-law behavior only over a certain range of degree. Taking the minimum of this range to be $k = 1$ and denoting the maximum by $k_{\max}$, we have

$$\frac{\langle k^2 \rangle}{\langle k \rangle} \sim -\frac{k_{\max}^{3-\gamma} - 1}{k_{\max}^{2-\gamma} - 1}. \tag{5.37}$$

Typical values of the exponent $\gamma$ fall in the range $2 < \gamma < 3$, so that $k_{\max}^{2-\gamma}$ is small for large $k_{\max}$ and can be ignored. On the other hand, $k_{\max}^{3-\gamma}$ becomes large in the same limit and hence $\langle k^2 \rangle / \langle k \rangle \sim k_{\max}^{3-\gamma}$ and

$$\tau \sim n k_{\max}^{\gamma-3}. \tag{5.38}$$

The scaling of the search time with system size $n$ thus depends, in this case, on the scaling of the maximum degree $k_{\max}$.

As an example, Aiello *et al.* [4] studied power-law degree distributions with a cut-off of the form $k_{\max} \sim n^{1/\gamma}$, which gives

$$\tau \sim n^{2-3/\gamma}. \tag{5.39}$$

A similar result was obtained previously by Adamic *et al.* [3] using different methods.

### 5.4.4 Bandwidth

Bandwidth is the mean number of queries reaching a given vertex per unit time. Equation (5.33) tells us that the probability that a particular current query reaches vertex $i$ at a particular time is $k_i/2m$, and assuming as discussed above that the number of queries initiated per unit time is proportional to the total number of vertices, the bandwidth for vertex $i$ is

$$Bn\frac{k_i}{2m} = B\frac{k_i}{\langle k \rangle}, \tag{5.40}$$

where $B$ is another constant.

This implies that high-degree vertices will be overloaded by comparison with low-degree ones so that, despite their good performance in terms of search times, networks with power-law or other highly right-skewed degree distributions may be undesirable in terms of bandwidth, with bottlenecks forming around the vertices of highest degree that could harm the performance of the entire network. If we wish to distribute load more evenly among the computers in our network, a network with a tightly peaked degree distribution is desirable.

### 5.4.5 Choice of network

A simple and attractive choice for our network is the Poisson distributed network of Section 5.3. For a Poisson degree distribution with mean $\mu$ we have $\langle k \rangle = \mu$ and $\langle k^2 \rangle = \mu^2 + \mu$. Then, using Eq. (5.36), the average search time is

$$\tau = A\frac{n}{\mu}. \tag{5.41}$$

As we have seen, a network of this type can be realized in practice with a biased-random-walker attachment mechanism of the kind described in Section 5.3.

Now if we allow $\mu$ to grow as some power of the size of the entire network, $\mu \sim n^\alpha$ with $0 \le \alpha \le 1$, then $\tau \sim n^{1-\alpha}$. For smaller values of $\alpha$, searches will take longer, but vertices' degrees are lower on average meaning that each vertex will have to devote less memory resources to maintaining its directory. Conversely, for larger $\alpha$, searches will be completed more quickly at the expense of greater memory usage. In the limiting case $\alpha = 1$, searches are completed in constant time, independent of the network size, despite the simple-minded nature of the random walk search algorithm.

The price we pay for this good performance is that the network becomes dense, having a number of edges scaling as $n^{1+\alpha}$. It is important to bear in mind, however, that this is a *virtual* network, in which the edges are a purely notional construct whose creation and maintenance carries essentially zero cost. There *is* a cost associated with the directories maintained by vertices, which for $\alpha = 1$ will contain information on the items held by a fixed fraction of all the vertices in the network. For instance, each vertex might be required to maintain a directory of 1% of all items in the network. Because of the nature of modern computer technology, however, we don't expect this to create a significant problem. User time (for performing searches) and CPU time and bandwidth are scarce resources that must be carefully conserved, but memory space on hard disks is cheap, and the tens or even hundreds of megabytes needed to maintain a directory is considered in most cases to be a small investment. By making the choice $\alpha = 1$ we can trade cheap memory resources for essentially optimal behavior in terms of search time and this is normally a good deal for the user.

We note also that the search process is naturally parallelizable: there is nothing to stop the vertex originating a search from sending out several independent random walkers and the expected time to complete the search will be reduced by a factor of the number of walkers. Alternatively, we could reduce the degrees of all vertices in the network by a constant factor and increase the number of walkers by the same factor, which would keep the average search time constant while reducing the sizes of the directories substantially, at the cost of increasing the average bandwidth load on each vertex.

As a test of our proposed search scheme, we have performed simulations of the procedure on Poisson networks generated using the random-walker method of Section 5.3. Figure 5.2 shows as a function of network size the average time $\tau$ taken by a random walker to find an item placed at a single randomly chosen vertex in the

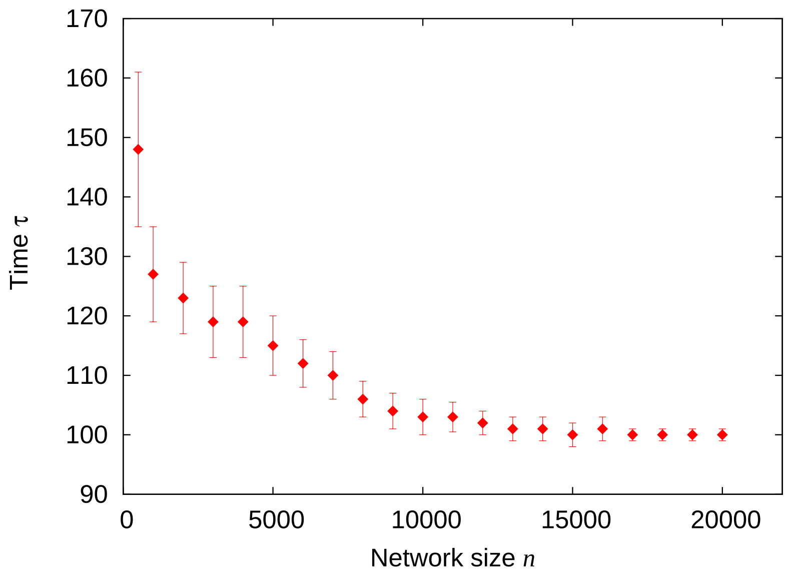

Figure 5.2: The time $\tau$ for the random walk search to find an item deposited at a random vertex, as a function of the number of vertices $n$.

network. As we can see, the value of $\tau$ does indeed tend to a constant (about 100 steps in this case) as network size becomes large.

We should also point out that for small values of $\mu$ vertices with degree zero could cause a problem. A vertex that loses all of its edges because its neighbors have all left the network can no longer be reached by our random walkers, and hence no vertices can attach to them and our attachment scheme breaks down. However, in the case considered here, where $\mu$ becomes large, the number of such vertices is exponentially small, and hence they can be neglected without substantial deleterious effects. Any vertex that does find itself entirely disconnected from the network can simply rejoin by the standard mechanism.

### 5.4.6 Item frequency distribution

In most cases, the search problem posed above is not a realistic representation of typical search problems encountered in peer-to-peer networks. In real networks, copies of items often occur in many places in the network. Let $s$ be the number of times a particular item occurs in the network and let $p_s$ be the probability distribution of $s$ over the network, i.e., $p_s$ is the fraction of items that exist in $s$ copies.

If the item we are searching for exists in $s$ copies, then Eq. (5.41) becomes

$$\tau_s = A\frac{n}{\mu s}, \tag{5.42}$$

since the chance of finding a copy of the desired item is multiplied by $s$ on each step of the random walk. On the other hand, it is likely that the frequency of searches for items is not uniformly distributed: more popular items, that is those with higher $s$, are likely to be searched for more often than less popular ones. For the purposes of illustration, let us make the simple assumption that the frequency of searches for a particular item is proportional to the item's popularity. Then the average time taken by a search is

$$\langle\tau\rangle = \frac{\sum_{s=1}^{\infty} s p_s \tau_s}{\sum_{s=1}^{\infty} s p_s} = A\frac{n}{\mu\langle s\rangle}, \tag{5.43}$$

where we have made use of $\sum_s p_s = 1$ and $\sum_s s p_s = \langle s\rangle$.

One possibility is that the total number of copies of items in the network increases in proportion to the number of vertices, but that the number of *distinct* items remains roughly the same, so that the average number of copies of a particular item increases as $\langle s\rangle \sim n$. In this case, $\langle\tau\rangle$ becomes independent of $n$ even when $\mu$ is constant, since we have to search only a constant number of vertices, not a constant fraction, to find a desired item. Perhaps a more realistic possibility is that the number of distinct items increases with network size, but does so slower than $n$, in which case one can achieve constant search times with a mean degree $\mu$ that also increases slower than $n$, so that directory sizes measured as a fraction of the network size dwindle.

An alternative scenario is one of items with a power-law frequency distribution $p_s \sim s^{-\delta}$. This case describes, for example, most forms of mass art or culture including books and recordings, emails and other messages circulating on the Internet, and many others [109]. The mean time to perform a search in the network then depends on the value of the exponent $\delta$. In many cases we have $\delta > 2$, which means that $\langle s \rangle$ is finite and well-behaved as the database becomes large, and hence $\langle \tau \rangle$, Eq. (5.43), differs from Eq. (5.41) by only a constant factor. (That factor may be quite large, making a significant practical difference to the waiting time for searches to complete, but the scaling with system size is unchanged.) If $\delta < 2$, however, then $\langle s \rangle$ becomes ill-defined, having a formally divergent value, so that $\langle \tau \rangle \to 0$ as system size becomes large. Physically, this represents the case in which most searches are for the most commonly occurring items, and those items occur so commonly that most searches terminate very quickly.

While this extra speed is a desirable feature of the search process, it's worth noting that average search time may not be the most important metric of performance for users of the network. In many situations, worst-case search time is a better measure of the ability of the search algorithm to meet users' demands. Assuming that the most infrequently occurring items in the network occur only once, or only a fixed number of times, the worst-case performance will still be given by Eq. (5.41).

### 5.4.7 Estimating network size

One further detail remains to be considered. If we want to make the mean degree $\mu$ of vertices added to the network proportional to the size $n$ of the entire network, or to some power of $n$, we need to know $n$, which presents a challenge since, as we have said, we do not expect any vertex to know the identity of all or even most of the other the vertices. This problem can be solved using a breadth-first search, which can be implemented once again by message passing across the network. One vertex $i$ chosen at random (or more realistically every vertex, at random but stochastically constant intervals proportional to system size) sends messages to some number $d$ of randomly chosen neighbors. The message contains the address of vertex $i$, a unique identifier string, and a counter whose initial value is zero. Each receiving vertex increases the counter by 1, passes the message on to one of its neighbors, and also sends messages with the same address, identifier, and with counter zero to $d-1$ other neighbors. Any vertex receiving a message with an identifier it has seen previously sends the value of the counter contained in that message back to vertex $i$, but does not forward the message to any further vertices. If vertex $i$ adds together all of the counter values it receives, the total will equal the number of vertices (other than itself) in the entire network. This number can then be broadcast to every other vertex in the network using a similar breadth-first search (or perhaps as a part of the next such search instigated by vertex $i$.)

The advantage of this process is that it has a total bandwidth cost (total number of messages sent) equal to $dn$. For constant $d$ therefore, the cost per vertex is a constant and hence the process will scale to arbitrarily large networks without consuming bandwidth. The (worst-case) time taken by the process depends on the longest geodesic path between any two vertices in the network, which is $\mathrm{O}(\log n)$. Although not as good as $\mathrm{O}(1)$, this still allows the network to scale to exponentially large sizes before the time needed to measure network size becomes an issue, and it seems likely that directory size (which scales linearly with or as a power of $n$ depending on the precise algorithm) will become a limiting factor long before this happens.

## 5.5 Discussion

In this chapter, we have considered the problem of designing networks indirectly by manipulating the rules by which they evolve. For certain types of networks, such as peer-to-peer networks, the limited control that this manipulation gives us over network structure, such as the ability to impose an arbitrary degree distribution of our choosing on the network, may be sufficient to generate significant improvements in network performance. Using generating function methods, we have shown that it is possible to impose a (nearly) arbitrary degree distribution on a network by appropriate choice of the "attachment kernel" that governs how newly added vertices connect to the network. Furthermore, we have described a scheme based on biased random walks whereby arbitrary attachment kernels can be implemented in practice.

We have also considered what particular choices of degree distribution offer the best performance in idealized networks under simple assumptions about search strategies and bandwidth constraints. We have given general formulas for search times and bandwidth usage per vertex and studied in detail one particularly simple case of a Poisson network that can be realized in straightforward fashion using our biased random walker scheme, allows us to perform decentralized searches in constant time, and makes only constant bandwidth demands per vertex, even in the limit where the database becomes arbitrarily large. No part of the scheme requires any centralized knowledge of the network, making the network a true peer-to-peer network, in the sense of having client nodes only and no servers.

One important issue that we have neglected in our discussion is that of "supernodes" in the network. Because the speed of previous search strategies has been recognized as a serious problem for peer-to-peer networks, designers of some networks have chosen to designate a subset of network vertices (typically those with above-average bandwidth and CPU resources) as supernodes. These supernodes are themselves connected together into a network over which all search activity takes place. Other client vertices then query this network when they want a search performed. Since the size of the supernode network is considerably less than the size of the network as a whole, this tactic increases the speed of searches, albeit only by a constant factor, at the expense of heavier load on the supernode machines. It would be elementary to generalize our approach to incorporate supernodes. One would simply give each supernode a directory of the data items stored by the client vertices of its supernode neighbors. Then searches would take place exactly as before, but on the supernode network alone, and client vertices would query the supernode network to perform searches. In all other respects the mechanisms would remain the same.

CHAPTER VI

# The Diplomats Dilemma: A Game theory model of Social Networks

## 6.1 Introduction

We now come to the final topic of the dissertation. The material covered so far, has primarily dealt with using mathematical methods to try and explain the observed properties (both structural and dynamic) of real networks.

Here, we switch gears and instead use computer modeling to simulate dynamical processes in networks. In particular, we study in detail an example of what is referred to as an *adaptive network*. In adaptive networks, there is a direct link between the structure as well as function, in the sense that one is directly dependent on the other and vice-versa. The system that we study has been inspired by a rather common situation in society. As motivation consider the following.

In many walks of life, particularly in situations involving conflict or competition, it is a goal of individuals to aspire to a position of power or influence. Moreover, once these individuals have attained this goal, it is reasonable to suppose that they will try and expend a considerable amount of effort to stay in that position of influence. A good example of this is provided by a network of diplomats or political lobbyists, who as we know seek to make connections to important folks, such that they can lobby for the interests they serve. Now, at this point we must come up with a reasonable definition of what we mean by "power" or "influence" and as it turns out, we are not the first to ask this question. Over the past century many sociologists have tried to define this concept, and although there are varied suggestions, most agree that it not an intrinsic property of an individual or actor[1], but rather a result of interactions between people. One well-known definition by the German polymath Max Weber, reads [150]:

> 'Power' is the probability that one actor within a social relationship will be in position to carry out his own will despite resistance, regardless of the basis on which this probability rests.

Definitions like this suggest that there is a link between the power of an actor and its position in a network of social relationships. It seems reasonable, then, that if we were to study the dynamics of a a social network, we should be able to say something about the power of the agents, or indeed, their efficacy in exercising that privilege. Consequently, a major theme in the study of social networks has been to discern the power structures in organizations based on the contact patterns of their members [92]. In a network of actors, coupled pairwise by their social ties, one idea of measuring, or defining power is to say that an actor that is closer to others has greater influence than a more peripheral one [133]. Closeness in this context, refers to the distance between two individuals in a network. So for example, immediate neighbors are one's closest friends, friends of friends are the next closest, and so on. One way to measure this definition of friendship or influence is to employ the *closeness centrality* that we defined in Sec. 1.4.5.

Naively then, one way to achieve a degree of primacy in a social network would be to position oneself as close to everyone else as possible, i.e. to have a social tie

[1]A person, or other well-defined social unit. In the context of our model we will use the term *agent*.

to each one of the network's actors. However in practical terms, establishing and maintaining a social tie requires an individual to invest a significant amount of effort in terms of time and other resources. To have a direct tie to an extensive fraction of the network is thus neither feasible, nor desirable. We call this situation of two competing interests—to maximize power (in terms of being central), while at the same time keeping the number of social ties to a minimum—the *diplomat's dilemma*.

Traditionally problems of this type have been studied under the aegis of *game theory*. Roughly speaking, game theory is a branch of applied mathematics that attempts to capture the behavior in strategic situations where an individual's success depends on his/her performance in relation to the other participants in the game. Though initially developed [146] to study situations where a participant does well at the expense of another, so called *zero-sum* games, it has been expanded to include many different types of criteria, such as co-operative and non-co-operative games, symmetric and non-symmetric, simultaneous and sequential, among many others [14]. The connection between networks and game theory has been precipitated by the increased availability of large-scale data-sets of a diverse nature ranging from socio-economic data (the traditional subject of scrutiny in game theory) to technological and biological data. On the one hand this had led to a discovery of new interactions and processes that can be scrutinized through the lens of game theory; on the other hand the sheer scale of the data, along with the powerful computational resources available to researchers nowadays, makes it well suited to be represented and studied as a network. For an extensive review of research combining aspects of network analysis and game theory, see the book by Matthew Jackson [82].

Some of the more interesting game-theoretical problems have been inspired by situations where the agents have conflicting objectives. In, for example, the iterated prisoner's dilemma [15], agents have to choose between trying to achieve short-term benefits by exploiting other agents, and trying to optimize their long-term profit (referred to in the literature as *payoff*) by building a relationship of mutual trust, but at the same time making them vulnerable to exploitation. In the spirit of competing interests, then, a potentially interesting question in the interface between complex networks and game theory would be: How can agents simultaneously maximize their centrality and minimize their degree?

In this chapter, we develop and examine a model of adaptive agents that try to solve this problem as the network evolves [76] in response to their decisions. We then examine in detail the output of this model, in terms of both the evolution of the network as well as that of the strategies of the agents. We note that in most models of a similar nature [72], the performance of the agents is related to some exogenous traits assigned to the agents. For example, Jackson and Wolinsky [83] have studied a more general version of this problem, where in addition to a distance measure between vertices, they consider the value and cost of maintaining connections between agents, which they define by means of an *utility function* that keeps track of a vertex's performance in the game. However, their main focus is in identifying stable structures—given the conditions of the game (which are stringent)—that also lead to efficiency, in the sense of maximizing an individual agent's utility function. Our model differs from this approach in that the success of our agents can be measured only from the topological features of the network rather than some *extremal* attribute artificially ascribed to the agents. Moreover, the stable structures they identify are comparatively trivial—star graphs for example. By defining our analog of the utility function based only on an agent's position in the network, we uncover far richer dynamical and topological variations of the network.

## 6.2 Definition of the Model

### 6.2.1 Preliminaries

The template for our study is a graph $G(t) = \{V, E(t)\}$ of $n$ vertices $V$, and $m(t)$ edges $E(t)$. The vertex set $V$ is fixed, but the edge set $E(t)$ is allowed to vary (both in its configuration and size) as a function of time. The vertices represent the agents in our network, while the edges mark the social ties between them.

Let $d_{ij}$ denote the shortest distance (geodesic) between two agents $i$ and $j$, i.e, the smallest sequence of adjacent edges connecting them. Then for a connected graph $G$—a network where any two pairs of vertices are connected by some path—the closeness centrality (see Sec. 1.4.5) for a vertex $i$ is defined as:

$$c_c(i) = \frac{n-1}{\sum_{j \neq i} d_{ij}}, \tag{6.1}$$

where $n$ is the number of vertices in the graph. In other words, it is the reciprocal of the average geodesic distance from a vertex $i$ to all other vertices in the network. Thus, smaller distances lead to a large centrality measure, whereas larger distances of $\mathrm{O}(n)$ make a negligible contribution.

In order to gauge the success of our agents, we will need some measure or score function, that takes into account our conflicting objects of high centrality and low degree. The simplest choice for such a function is $c_c(i)/k_i$ (where $k_i$ represents the degree of vertex $i$), since it decreases with the number of ties, and increases with centrality. However, we do not want to restrict ourselves to connected networks. If the network is split up into disconnected components, we would like to impose the condition, that a vertex that belongs to a larger component of the network, has a larger contribution to its centrality (since a vertex that is connected to a thousand other vertices is likely to be more central or influential than one connected to just ten). One way of modifying our definition of closeness centrality, such that both small distances as well as membership in a large component contribute positively, is to redefine it as:

$$c(i) = \sum_{j \neq i} \frac{1}{d_{ij}}. \tag{6.2}$$

The number of elements in the sum of Eq. (6.2) is proportional to the number of vertices of $i$'s connected component (as vertices that are disconnected have $d_{ij} = \infty$ which makes zero contribution to the sum) and thus large components make a strong positive contribution. Equation (6.2) measures the average of the reciprocal distance, rather than the reciprocal of the average distance (as in the original definition of closeness centrality). This adjusted definition assigns a higher weight on the count of closer vertices, but nevertheless captures similar features as the traditional version shown in Eqn. (6.1).

With the definitions established above, we are now in a position to state our score function:

$$s(i) = \begin{cases} c(i)/k_i & \text{if } k_i > 0 \\ 0 & \text{if } k_i = 0 \end{cases}, \tag{6.3}$$

where the condition for $k_i = 0$ naturally follows, since a vertex with no connections is not likely to be central to anyone but itself.

Before we move on, we must decide on a choice of network on which to run our game. For the sake of simplicity, our starting network will be an Erdős-Rényi network [56] with $m_0$ number of initial edges, where the network is generated by adding $m_0$ edges one-by-one to $n$ (isolated) vertices such that no multiple or self-edges are formed (in the context of our model it is unclear what such an edge would mean). This is just the $G_{n,m}$ model discussed in the first Chapter, see Sec. 1.5.1. Other

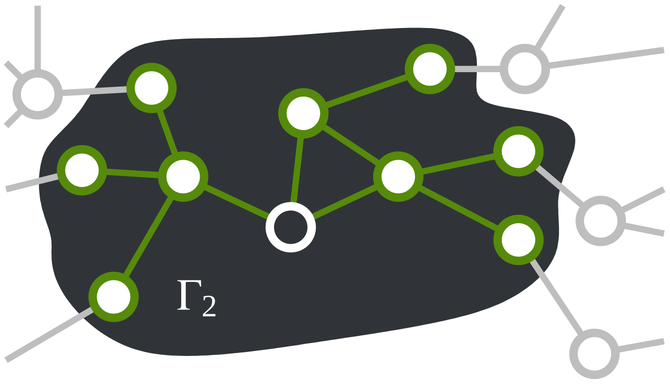

Figure 6.1: An illustration of the myopia (the restricted knowledge of the network). The agents are assumed to have knowledge of, and be able to affect the second neighborhood $\Gamma_2$ (shaded in the figure). Each agent is aware of the centrality and degree of its neighbors and their accumulated scores. Based on this information the agents can, during a time step, based on their strategies, decide to delete an edge to a neighbor, and reconnect to a vertex two steps away.

complicated choices are certainly possible, however as we will see, the dynamical evolution of the model is sufficiently rich, such that even this simple choice leads to very striking results.

### 6.2.2 Moves and Strategies

We have outlined so far the basic setup for the game—the underlying graph representing the actors and their social network, and the score function that the agents want to optimize. However, to go from this point to a sensible simulation scheme, we need to determine how an agent can update its connections.

To begin with, we will assume that the agents are *myopic*—they can receive information from, and affect others in the network, only within a fixed neighborhood. This assumption is fairly common to much of the studies dealing with social networks and is probably a fairly accurate assessment of situations in the real world as well (a physicist working at the physics department in a university is more likely to know members of the same department rather than someone from the history department.)

Based on this property of myopia, we assume that an agent $i$ can change connections (affect the network) only within its second neighborhood $\Gamma_2 = \{j \in V : d_{ij} \leq 2\}$, and that $i$ can see the score $s(j)$, centrality $c(j)$ and degree $k_j$ of vertices in $\Gamma_2$. Note that $s$ and $c$ are global quantities, therefore some global information reaches $i$ indirectly. Nevertheless, since the actual contact network cannot be inferred from this information, we still consider the agents myopic. See Fig. 6.1 for an illustration.

Ideally one would like to provide the agents with some intelligence and use no further restrictions for how they update their positions to increase their scores. This is however a rather difficult proposition from a simulation point of view, and therefore we would have to impose some additional (sensible) restrictions for purposes of tractability. We will assume the agents can change their neighborhood through only two ways—attaching an edge to a new vertex, or deleting an edge from an existing neighbor. In terms of a friendship network, one can think of the first action as making new friends for profit, and the second action as unceremoniously dumping old ones should they not prove useful to one's objectives (a rather common phenomena in the real world!).

Despite the simplicity of the moves themselves, the criteria an agent applies to decide on a particular move, can be made quite complex. We will assume that an agent $i$ updates its position (either by deleting or attaching an edge), by applying a sequence of tie-breaking *actions* chosen from the following set:

- **MAXD** Choose vertices with maximal degree.
- **MIND** Choose vertices with minimal degree.
- **MAXC** Choose vertices with maximal centrality in the sense of Eq. (6.2).
- **MINC** Choose vertices with minimal centrality.

- **RND** Pick a vertex at random.
- **NO** Do not add (or remove) any edge.

The sequences of actions define the *strategies* of the agents. The strategy of an agent $i$ can be stored in two six-tuples $\mathbf{s}_{\text{add}} = (s_1^{\text{add}}, \cdots, s_6^{\text{add}})$ and $\mathbf{s}_{\text{del}} = (s_1^{\text{del}}, \cdots, s_6^{\text{del}})$ representing a priority ordering of the addition and deletion actions respectively.

If for example, $\mathbf{s}_{\text{add}}(i) = (\text{MAXD}, \text{MINC}, \text{NO}, \text{RND}, \text{MIND}, \text{MAXC})$ then $i$ tries at first to attach an edge to the vertex in $\Gamma_2(i)$ with highest degree. If more than one vertex has the highest degree, then one of these is selected by the MINC strategy. If still no unique vertex is found, nothing is done (by application of the NO strategy). Note that such a vertex is always found after strategies NO or RND are applied. If the neighborhood set of a vertex is $X = \varnothing$, no edge is added (or deleted).

The simulations proceed iteratively where at each time step, every vertex can update its network position by adding an edge to a vertex in $\Gamma_2$ and delete an edge to a neighbor. An example of the possible moves and strategies available to an agent is shown in Fig. 6.2.

### 6.2.3 Strategy updates and stochastic rewiring

The strategy vectors are initialized to random permutations of the six actions. Every $t_{\text{strat}}$th time step an agent $i$ updates its strategy vectors by finding the vertex in $\Gamma_i = \{j : d_{ij} \leq 1\}$ with the highest accumulated score since the last strategy update. This practice of letting the agent mimic the best-performing neighbor is common in spatial games [120], and is closely related to the bounded rationality paradigm of economics [84], which is also an inspiration for the myopic property of the agents. When updating the strategy, $i$ copies the parts of $\mathbf{s}_{\text{add}}(j)$ and $\mathbf{s}_{\text{del}}(j)$ that $j$ employed in the previous time step, and let the remaining actions come in the same

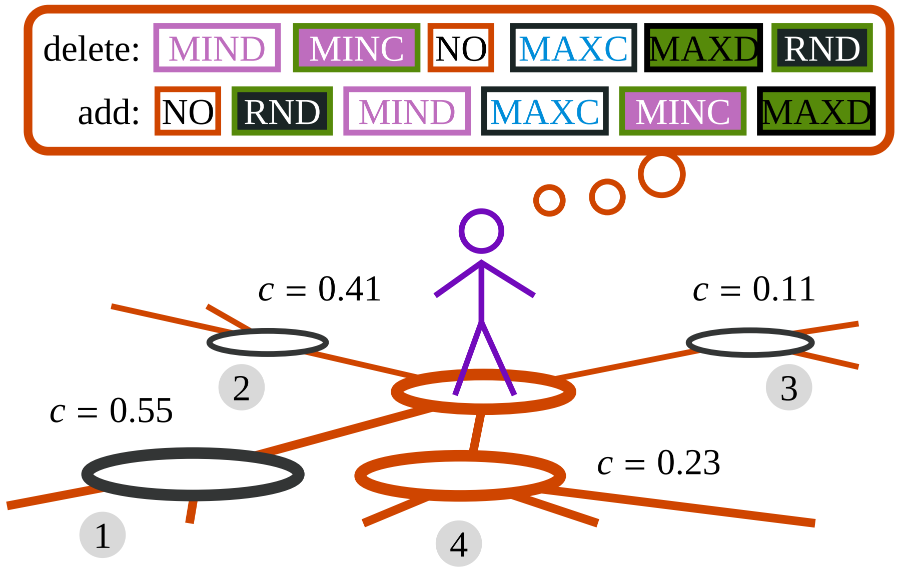

Figure 6.2: An illustration of the strategies employed by the agents. At a given time step, an agent can delete one edge and add another in order to improve its score. The way to select a neighbor to delete an edge to (or a next-nearest neighbor to attach an edge to) is to consecutively omit possibilities by applying "actions" in a "strategy vector". In the cartoon above, the agent's leading deletion strategy is MIND, meaning its initial preference is to sever ties to neighbors with the lowest degree. In this example there are three neighbors with degree three (marked in black). To further eliminate neighbors the agent applies the MINC strategy (ranking the neighbors in order of minimum centrality $c$. In this case vertex 3 is the least central neighbor. So, at this time step, the agent will delete the edge to 3. As for addition of edges the leading action is NO, meaning no edge will be added.

order as its own strategy vectors prior to the update. For the purposes of making the set of strategy vectors ergodic, driving the strategy optimization [121, 99], and modeling irrational moves by the agents [84]; we swap, with probability $p_s$, two random elements of $\mathbf{s}_{\text{add}}(j)$ and $\mathbf{s}_{\text{del}}(j)$ every strategy vector update.

In addition to the strategy space we also would like to impose ergodicity in the network space (i.e. the game can generate all $n$-vertex graphs from any initial configuration). In order to ensure this, disconnected clusters should have the ability to reconnect to the graph. We allow this by letting a vertex $i$ attach to any random vertex of $V$ with probability $p_r$ every $t_{\text{rnd}}$th time step. This is not unreasonable as even in real social systems, edges may form between agents out of sight from each other in the social network. As many authors have pointed out, in addition to information spreading processes, there are other factors that lead to the evolution of the social networks (cf. Ref. [149]).

### 6.2.4 The algorithm

To summarize, the algorithm works as follows:

1. Initialize the network to a Erdős-Rényi network with $n$ vertices and $m_0$ edges.
2. For all agents, start with random permutations of the six actions as strategy vectors $\mathbf{s}_{\text{add}}$ and $\mathbf{s}_{\text{del}}$.
3. Calculate the score for all agents.
4. Update the agents synchronously by adding and deleting edges as selected by the strategy vectors. With probability $p_r$, add an edge to a random vertex instead of a neighbor's neighbor.
5. Every $t_{\text{strat}}$th time step, update the strategy vectors. For each agent, with probability $p_s$, swap two elements in its strategy vector.
6. Increment the simulation time $t$. If $t < t_{\text{tot}}$, go to step 3.

For the purposes of our simulation, we will use the parameter values $m_0 = 3n/2$ (for different values of $n$), $p_s = 0.005$, $t_{\text{strat}} = 10$, $t_{\text{tot}} = 10^5$ and $n_{\text{avg}} = 100$, where $n_{\text{avg}}$ denotes the average of quantities over multiple realizations of the network.

## 6.3 Numerical results

### 6.3.1 Time evolution

As a first look at the time evolution, we start by plotting quantities characterizing the strategies of the agents and the network structure. The most important parts of the strategy vectors are its first components, $s_1^{\text{add}}$ and $s_1^{\text{del}}$. In practice, $\sim 90\%$ of the decisions–whether or not to add (or delete) a specific edge—do not pass this first tiebreaker. In Fig. 6.3(a) and (b) we can see how complex the time-evolution of $s_1^{\text{add}}$ and $s_1^{\text{del}}$ can be. Each sector of the plot corresponds to a leading addition (or deletion) action. The vertical axis is a measure of the fraction of agents having that leading action value. The time evolution is complex, having sudden cascades of strategy changes and quasi-stable periods. Cascades in the leading addition action seem to be accompanied by cascades in the leading deletion action. For the parameter values of Fig. 6.3, cascades involving more than 75% of the vertices happens about once every $10^5$ time steps.

In Fig. 6.3(c) we measure the average score function $\langle s \rangle$ as a function of time. Being a non-zero-sum game, the value of $\langle s \rangle$ can vary significantly, a fact which can be seen upon examining the figure. Most of the time, the system is close to the observed maximum $\langle s \rangle \approx 80$. A possible explanation for the periods with lower scores can be seen in Fig. 6.3(d) where we plot the average degree $\langle k \rangle$. For some time steps, the network becomes very dense with an average degree of almost 20. As high degree is counter-productive to the objectives of the game, the average score is low during this period. The rise in degree has, naturally, a corresponding peak in the leading deletion action NO, since as a consequence of not dropping links, the agents accumulate a large number of neighbors.

Another reason for the occasional dips in the average score can be seen in Fig. 6.3(e) where we plot the fraction of agents, $n_1$, that belong to the largest connected component. This quantity is usually close to one, meaning that all agents are connected (directly or indirectly), however as can be seen from the figure, there are periods when the fraction drops substantially. The corresponding strategic cause for these

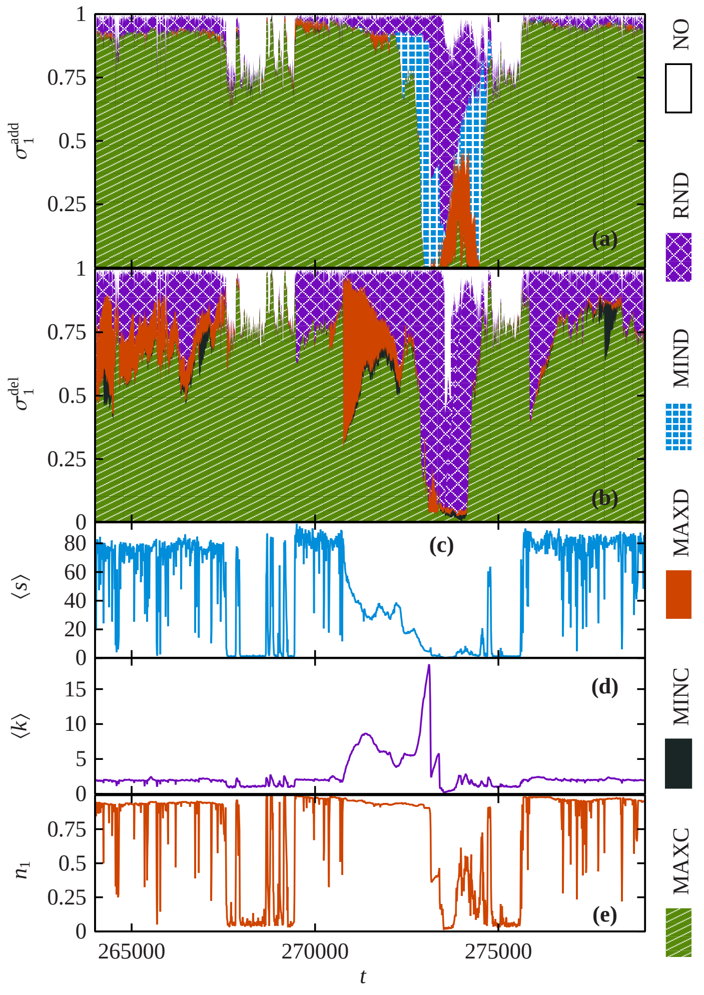

Figure 6.3: Output from an example run of a $n = 200$ system with $p_r = 0.012$. Panels (a) and (b) show the fraction of vertices having a particular leading action for addition $\sigma_1^{\text{add}}$ and deletion $\sigma_1^{\text{add}}$ respectively. Panel (c) shows the average score $\langle s \rangle$, (d) the average degree $\langle k \rangle$ and (e) the fraction of vertices in the largest connected component $n_1$.

fragmented states is not immediately obvious. They seem to correspond to a state of inactivity—the leading action for both deletion and addition is NO—however, as we will see, this feature becomes less pronounced with increasing system size.

### 6.3.2 Example networks

In light of the complex time evolution of the system, it is not surprising that the system attains a great variety of network topologies as time progresses. In Fig. 6.4 we show four snapshots of the system for a run with the same parameter values

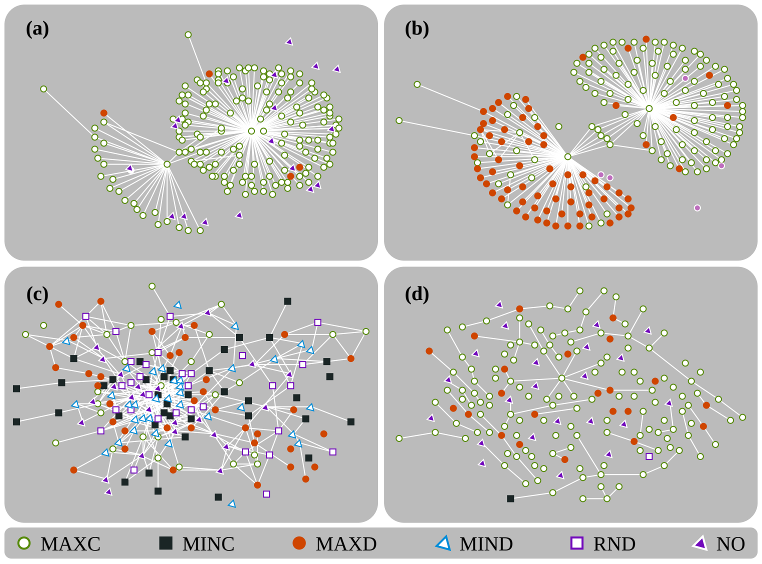

Figure 6.4: Four different example networks from a run with the same parameter values as in Fig. 6.3. The symbols indicate the leading addition action. (a) shows the common situation where MAXC is the leading addition action, $\sigma_1^{\text{del}}$ is MAXC for almost all agents; (b) shows a transition stage between $\sigma_1^{\text{add}}$ being mostly MAXD to $\sigma_1^{\text{add}}$ being primarily MAXC; (c) shows another transient configuration where a large number of different addition strategies coexist; (d) shows the addition strategies in a fragmented state.

as in Fig. 6.3. In Fig. 6.4(a) the network is a result of the most common strategy configuration where both the leading deletion and addition actions are MAXC for a majority of the agents (in this situation, we call the actions *dominating*). In this configuration the network is centered around two indirectly connected hubs. The vertices between these two hubs have the highest centrality, and since they are within the second neighborhood of most vertices in the network, and most agents have $\sigma_1^{\text{add}} = \text{MAXC}$, these vertices will get an edge from the majority of agents (thus becoming hubs in the next time-step). There are 18 isolated agents with $\sigma_1^{\text{add}} = \text{NO}$. These will continue to stay isolated until their strategy vectors are mutated, which occurs (on average) every $t_{\text{strat}}/p_s = 2000$th time step.

Figure 6.4(b) shows a rather similar network topology with the difference that a majority of the vertices have MAXD as their leading addition action (almost all vertices have $\sigma_1^{\text{del}} = \text{MAXC}$). For this configuration, the MAXC vertices will move their edges to the most central vertices whereas the MAXD vertices will not move theirs. In Fig. 6.4(c) we show a rare, high-$\langle k \rangle$ configuration. Here the leading deletion action is NO for about a quarter of the agents, and consequently the system is rapidly accumulating edges.

InFig. 6.4(d) we show a fragmented state, where a number of vertices have the leading addition action NO. The vertices in the connected component with leading addition action $\sigma_1^{\text{add}} =$ NO also have corresponding leading deletion action $\sigma_1^{\text{del}} =$ NO, so they will not fragment the network further. On the other hand, the vertices with $\sigma_1^{\text{add,del}} =$ MAXC and $\sigma_1^{\text{add,del}} =$ MAXD have the potential to further split up the network.

### 6.3.3 Effects of strategies on the network topology

We now turn our attention to how the evolution of the strategies affects the network topology. To do so, we measure a variety of metrics describing the structure of networks, starting with averages of various quantities and then moving on to more complicated measure such as assortatitvity and the clustering coefficient (cf. Chap. I). We address each of these turn.

As a first glimpse, in Fig. 6.5 we plot the probability density functions of the different structural quantities shown in Figs. 6.3(c), (d) and (e) after averaging them over multiple realizations of networks each allowed to evolve for a period of $10^5$ time-steps. As the figure shows, these all have two peaks—one with low $\langle s \rangle$, $\langle k \rangle$ and $\langle n_1 \rangle$ values (where the network is fragmented, the number of edges small and the scores low), and another broader peak corresponding to a connected network with higher scores and more edges. Interestingly, the various leading actions are not completely localized to different peaks but spread out over the whole range. Another counter-intuitive observation is that there seems to be more agents with $\sigma_1^{\text{add}} =$ NO in the

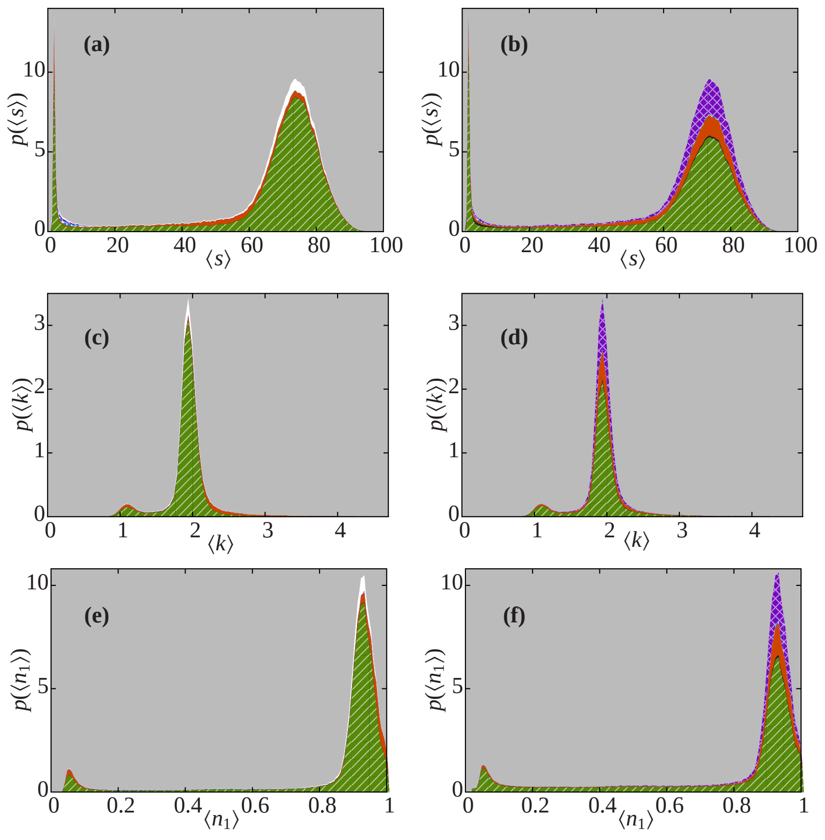

Figure 6.5: The probability density function of scores (a), (b), degrees (c), (d), and relative sizes of the largest connected component (e), (f) averaged over multiple realizations of the network, denoted by brackets $\langle \ldots \rangle$. The fields represent different leading addition actions (a), (c), (e), and different leading deletion actions (b), (d), (f). The vertical size of a field represents the probability density function conditioned to that leading action. The curves are averages over ten runs of $10^5$ timesteps with the same parameter values as in Fig. 6.3. The color codes of the actions are the same as in Fig. 6.3.

more dense peaks. These vertices (with $\sigma_1^{\text{add}} = \text{NO}$) seem to be primarily isolated and do not affect the majority of vertices (connected in the largest component). They will therefore stay isolated until their strategies have changed or they have been connected to the rest of the network by random connections. We also observe that there is a larger variety of leading addition actions than leading deletion actions. A possible interpretation of this is that the success of agents is more sensitive to their addition strategies rather than their deletion ones.

We next turn our attention to the the degree distribution $p_k$—the fraction of vertices in the network with exactly $k$ links. In Fig. 6.6 we plot the degree dis-

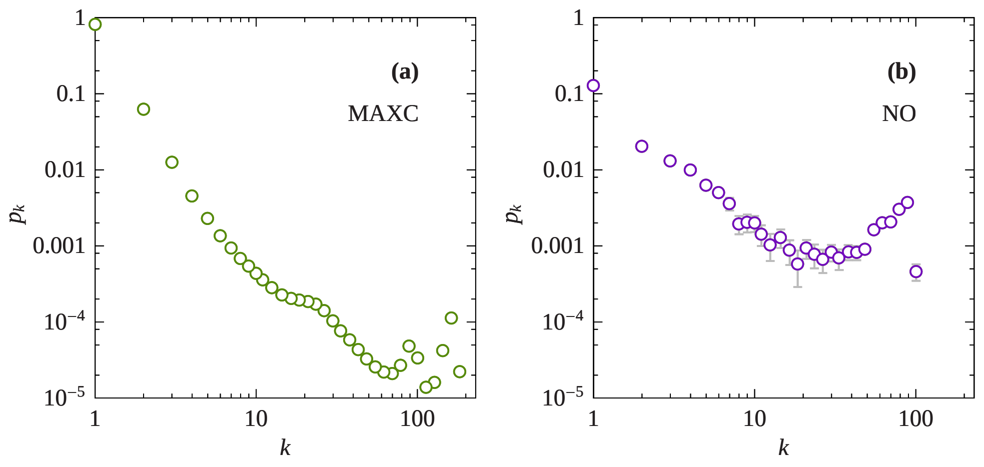

Figure 6.6: The degree distribution (log-log scale) for systems with the same parameter values as in Fig. 6.3. Panel (a) shows the averaged degree distribution when more than half of the agents have MAXC as their leading addition actions. Panel (b) displays the corresponding plot for the leading addition action NO.

tribution for agents (averaged over multiple realizations) with dominating actions $\sigma_1^{\text{add}} = \text{MAXC}$ (a) and $\sigma_1^{\text{add}} = \text{NO}$ (b). The $\sigma_1^{\text{add}} = \text{MAXC}$ graph has two high-$k$ peaks, corresponding to the hubs in the network. The existence of two broad peaks as opposed to only one is strange, and the reasons for this are not immediately apparent. The $\sigma_1^{\text{add}} = \text{NO}$ graphs are more dense, as expected. However, they also have a large-$k$ peak, which is probably related to, either the strategies of other agents, or a residue from the preceding period (remember that the periods of dominating $\sigma_1^{\text{add}} = \text{NO}$ are very short compared with the $\sigma_1^{\text{add}} = \text{MAXC}$ periods). This implies that one can separate system-wide effects of some strategy driving the decisions of the majority, but there will also be other effects present in the network.

We now proceed to look at the relation between the score $s$ and the degree $k$ of vertices for specific strategies employed by the agents. In Fig. 6.7(a) and (b) we plot the average score $\langle s \rangle$ as a function of the average degree $\langle k \rangle$ for each addition and deletion strategy respectively. The plot is the result of ten separate runs for $10^5$ time steps, with network quantities measured every tenth time step. During these runs, $\sigma_1^{\text{add}} = \text{MINC}$ and $\sigma_1^{\text{del}} = \text{MIND}$ were never employed as leading actions. It turns out that the most common strategies used by the agents (for both addition and deletion) MAXC and MAXD lead to the highest average score, although this does not necessarily imply that all agents are doing well—from Figs. 6.4(a) and (b) we know that the score can differ much from one agent to another.

The degrees are low for these strategies, which is a necessary (but not sufficient) condition for a low score. For $\sigma_1^{\text{add}} = \text{NO}$ the average degree is also low, but the score is much lower than for $\sigma_1^{\text{add}} = \text{MAXC}$ and MAXD. The reason, as pointed out above, is that the network can become heavily fragmented for this leading addition action. The addition strategy $\sigma_1^{\text{add}}$ corresponding to the highest degree is RND, which might seem strange, but during these runs (also visible in Fig. 6.3) $\sigma_1^{\text{add}} = \text{RND}$ is correlated with $\sigma_1^{\text{del}} = \text{NO}$ which is a state naturally leading to a comparatively dense network. The other leading actions $\sigma_1^{\text{add}} = \text{MIND}$ and $\sigma_1^{\text{del}} = \text{MINC}$ result in low scores and sparse networks.

Next we measure the assortativity and clustering coefficients. The assortativity $r$ is a measure of vertices' tendency to connect to other vertices of similar type, in this case those with similar degree [108]. In mathematical terms, $r$ is the Pearson correlation coefficient [129] of the degrees at either side of an edge. Before we measure this quantity we will need to make a slight modification. The edges in our networks are undirected, and therefore $r$ has to be symmetric with respect to edge-reversal (i.e. replacing $(i, j)$ by $(j, i)$). However the standard definition of the Pearson correlation coefficient does not account for this symmetry. An easy fix to this problem is to let one edge contribute twice to $r$, i.e. to represent an undirected edge by two directed edges pointing in opposite directions. If one employs an edge list representation (i.e., if edges are stored in an array of ordered pairs $(i_1, j_1), \cdots, (i_M, j_M)$) then we can

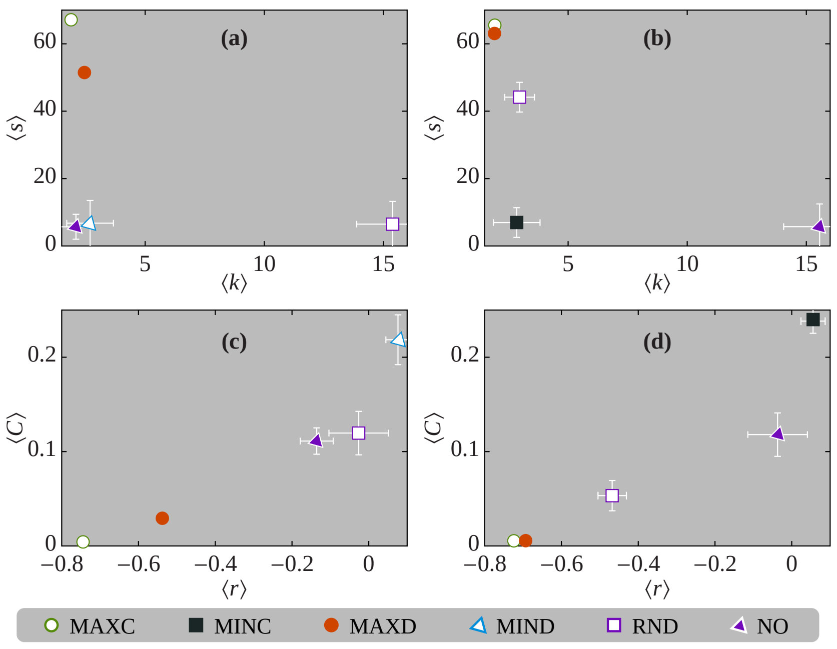

Figure 6.7: Average values of four different network structural quantities for different dominating addition and deletion actions (i.e. that more than half of the agents have a specific $\sigma_1^{\text{add}}$, or $\sigma_1^{\text{del}}$); (a) shows the average score as a function of degree for different dominating $\sigma_1^{\text{add}}$; (b) is the corresponding plot for $\sigma_1^{\text{del}}$; (c) displays the clustering coefficient as a function of assortativity for different dominating $\sigma_1^{\text{add}}$; (d) is the corresponding plot for different $\sigma_1^{\text{del}}$. The bars indicate standard errors. The data comes from simulation of ten runs (different random number generator seeds) of $10^5$ time steps. Two actions were never seen during these runs: $\sigma_1^{\text{add}} = \text{MINC}$ and $\sigma_1^{\text{del}} = \text{MIND}$.

write $r$ as,

$$r = \frac{4\langle k_1\, k_2\rangle - \langle k_1 + k_2\rangle^2}{2\langle k_1^2 + k_2^2\rangle - \langle k_1 + k_2\rangle^2}, \tag{6.4}$$

where, for a given edge $(i, j)$, $k_1$ is the degree of the first argument (i.e., the degree of $i$), $k_2$ is the degree of the second argument and the brackets $\langle \cdots \rangle$ denote averaging. The range of $r$ is $[-1, 1]$ where negative values indicate a preference for highly connected vertices to attach to low degree vertices, and positive values imply that vertices tend to be attached to other vertices with degrees of similar magnitudes.

The clustering coefficient is a measure of transitivity in the network. In other words it checks whether neighbors of a node are also connected to each other (thus forming triangles). It is a well known empirical fact that social acquaintance networks have a strong tendency to form triangles [74] and it is therefore a worthwhile exercise to examine whether the networks generated by our model display this feature. There is in principle, more than one way to define the clustering coefficient. Here we employ the most commonly used one [17],

$$C = 3n_{\Delta} \Big/ n_{\text{triple}}, \tag{6.5}$$

where $n_{\Delta}$ is the number of triangles and $n_{\text{triple}}$ is the number of connected triples (subgraphs consisting of three vertices and two or three edges). The factor of three is included to normalize the quantity to the interval $[0,1]$.

In Fig. 6.7 (c) and (d) we plot the clustering coefficient as a function of the assortativity $r$ for different strategies, once again averaged over multiple networks. As the figure shows the most popular strategies $\sigma_1^{\text{add,del}}$ = MAXC and MAXD have the lowest $\langle C\rangle$ and $\langle r\rangle$ values. There is very likely a fairly simple explanation for this. Consider a situation where three agents are all connected to each other in the form of a triangle. The removal of any one edge between two agents does not lead to much of a loss as they are still connected to each other via the third member of the triangle. In a situation where edges are expensive, this kind of redundancy is not desired. For this reason, it seems natural that, on average, the most successful strategies MAXC and MAXD have few triangles.

A negative value for the assortativity—high degree agents with low degree ones as neighbors—is also a conspicuous feature of the examples shown in Figs. 6.4(a) and (b) (most vertices there are only connected to the two hubs, but the hubs are not connected to each other). For networks with a broad spectrum of degrees, it is known that $\langle C\rangle$ and $\langle r\rangle$ are relatively strongly correlated [78]. This is also true in Figs. 6.7(c) and (d) where the relationship between $\langle C\rangle$ and $\langle r\rangle$ is monotonically increasing. The network configurations with highest $\langle C\rangle$ and $\langle r\rangle$ are the ones with $\sigma_1^{\text{add}}$ = MIND

| | MAXC | MINC | MAXD | MIND | RND | NO |
|---|---|---|---|---|---|---|
| MAXC | 1 | 0.0164(3) | 0.0088(2) | 0.0107(4) | 0.0151(5) | 0.0010(0) |
| MINC | 0.0169(3) | 1 | 0.0113(6) | 0.036(2) | 0.025(2) | 0.0017(3) |
| MAXD | 0.0093(3) | 0.0104(7) | 1 | 0.0103(6) | 0.0206(9) | 0.0003(0) |
| MIND | 0.0115(4) | 0.030(2) | 0.0130(7) | 1 | 0.059(5) | 0.0020(2) |
| RND | 0.0157(5) | 0.024(2) | 0.020(1) | 0.064(5) | 1 | 0.0023(5) |
| NO | 0.0007(0) | 0.0031(2) | 0.0009(0) | 0.0036(2) | 0.0042(4) | 1 |

Table 6.1: Values for the **T** matrices (6.6) for addition strategies. ($T_{ij}$ is the deviation from the expected value in a model of random transitions given the diagonal values.) The values are averaged over 100 realizations of the algorithm. All digits are significant to one standard deviation. The parameter values are the same as in Fig. 6.3. Numbers in parentheses are the standard errors in units of the last decimal.

and $\sigma_1^{\text{del}}$ = MINC. Since these networks are both sparse and fragmented, some components must have a large number of triangles (probably close to being fully connected). The denser states, with $\sigma_1^{\text{add}}$ = RND and $\sigma_1^{\text{del}}$ = NO, have intermediate $\langle C\rangle$ and $\langle r\rangle$ values, meaning that the edges are more homogeneously spread out, similar to the network in Fig. 6.4(c).

### 6.3.4 Transition probabilities

From Fig. 6.3 it seems likely that the ability of one leading action to grow in the population depends on the other predominant strategies in the system. For example, $\sigma_1^{\text{add}}$ = RND dominates after a period of many agents employing $\sigma_1^{\text{add}}$ = MIND as the leading strategy. Consequently, it is worth asking the question: How does the probability of one leading action depend on the configuration at earlier time steps?

We investigate this qualitatively by calculating the "transition matrix" $\mathbf{T}'$ with elements $T'(s_1, s_1')$ that represent the probability of a vertex with the leading action $s_1$ to have the leading action $s_1'$ at the next time step. Note that $\mathbf{T}'$, is not a transition matrix in the sense of other physical models, as the dynamics are not fully determined by its elements. If that were the case (i.e. the current strategy is independent of the strategy adopted in the previous time step) we would have the relation $T'_{ij} = \sqrt{T'_i T'_j}$. To study the deviation from this null-model, we assume

| | MAXC | MINC | MAXD | MIND | RND | NO |
|---|---|---|---|---|---|---|
| MAXC | 1 | 0.0100(2) | 0.0131(4) | 0.0094(2) | 0.0266(3) | 0.0126(3) |
| MINC | 0.0098(2) | 1 | 0.0070(3) | 0.010(1) | 0.0105(4) | 0.0050(3) |
| MAXD | 0.0133(4) | 0.0067(3) | 1 | 0.0055(2) | 0.0124(3) | 0.0062(2) |
| MIND | 0.0087(2) | 0.011(1) | 0.0054(2) | 1 | 0.0101(2) | 0.0055(3) |
| RND | 0.0269(3) | 0.0094(4) | 0.0128(3) | 0.0083(2) | 1 | 0.0072(3) |
| NO | 0.0097(3) | 0.0076(3) | 0.0053(2) | 0.0078(3) | 0.0131(3) | 1 |

Table 6.2: Same as in Tab. 6.1 but for deletion, instead of addition, strategies.

the diagonal (i.e. the frequencies of the strategies) given, and calculate $\mathbf{T}$ with its elements defined by,

$$T_{ij} = T'_{ij}/\sqrt{T'_i T'_j}. \tag{6.6}$$

The values of $\mathbf{T}$ for the parameters defined in Fig. 6.3 are displayed in Tabs. 6.1 and 6.2. The off-diagonal elements have much lower values than 1 (the average values are 0.014 for addition strategies and 0.010 for deletion). This reflects the contiguous periods of one dominating action. Note that transitions between MAXC and RND are over-represented: $T'^{\rm del}_{\rm MAXC,RND} \approx T'^{\rm del}_{\rm RND,MAXC} \approx 0.027$, which is more than twice the value of any other off-diagonal element involving MAXC or RND. As another token of the problem's complexity, the matrix is not completely symmetric $T'^{\rm del}_{\rm RND,NO}$ is twice ($\sim 3$ standard deviations) as large as $T'^{\rm del}_{\rm NO,RND}$ meaning that it is easier for RND to invade a population with NO as a leading deletion action, than vice versa.

### 6.3.5 Dependence on system size and noise

In this section we investigate how the behavior of the system is affected by the number of agents $n$ participating in the game as well as the noise level (degree of randomness) in the deletion and attachment mechanism.

In Fig. 6.8, we tune the fraction of random attachments, $p_r$ for three system sizes. In panels (a)–(c) we show the fraction of leading addition actions among the agents $\langle\Sigma_1^{\rm add}\rangle$ (averaged over $\sim 100$ runs and $10^5$ time steps). The quantities $\Sigma_1^{\rm add,del}$

denotes the fraction of agents having a specific $\sigma_1^{\mathrm{add,del}}$. As observed in Fig. 6.3(a) the leading action is MAXC followed by MAXD and RND. The leading deletion actions, as seen in panels (d)–(f), are ranked similarly except that MAXD has a larger (and increasing) presence. If $p_r = 1$, then all actions are equally likely (they do not have any meaning—all strategies will result in random moves equal to $s_1^{\mathrm{add}} = s_1^{\mathrm{del}} = \mathrm{RND}$).

There are trends in the $p_r$-dependences of $\langle\sigma_1^{\mathrm{add}}\rangle$, but apparently no emerging discontinuity. This observation, (which also seems to hold for the $p_s$-scaling), that there is no phase transition for any parameter value governing the probability of random permutations in the strategy vectors, is an indication that the results above can be generalized to a large parameter range. We also note that, although the system has the opportunity to be passive (i.e. agents having $s_1^{\mathrm{add}} = s_1^{\mathrm{del}} = \mathrm{NO}$), this does not happen. This situation is reminiscent of the “Red Queen hypothesis” of evolution [145]—organisms need to keep evolving to maintain their fitness.

Next we look at the how the average degree $\langle k\rangle$ as well as the fraction of agents in the largest component, $n_1$, are affected by system size $n$ as well as $p_r$. The average degree, plotted in Fig. 6.3(g) monotonously increases with $p_r$. There is, however, a qualitative difference in the size scaling—for $p_r \lesssim 0.12$ the average degree increases with $n$, for $p_r \gtrsim 0.12$ this situation is reversed.

In Fig. 6.3(h) we plot the average largest-component size as a function of $p_r$ for different system sizes. This as well is a monotonously increasing function of both $p_r$ and $n$—larger randomness, or a larger system size, means higher $\langle n_1\rangle$. In most network models, a decreasing average degree implies a smaller giant component. For $p_r \gtrsim 0.12$, in our model, the picture is the opposite—as the system grows the giant component spans an increasing fraction of the network. This also means that the agents, on average, reach the twin goals of keeping the degree low and the graph

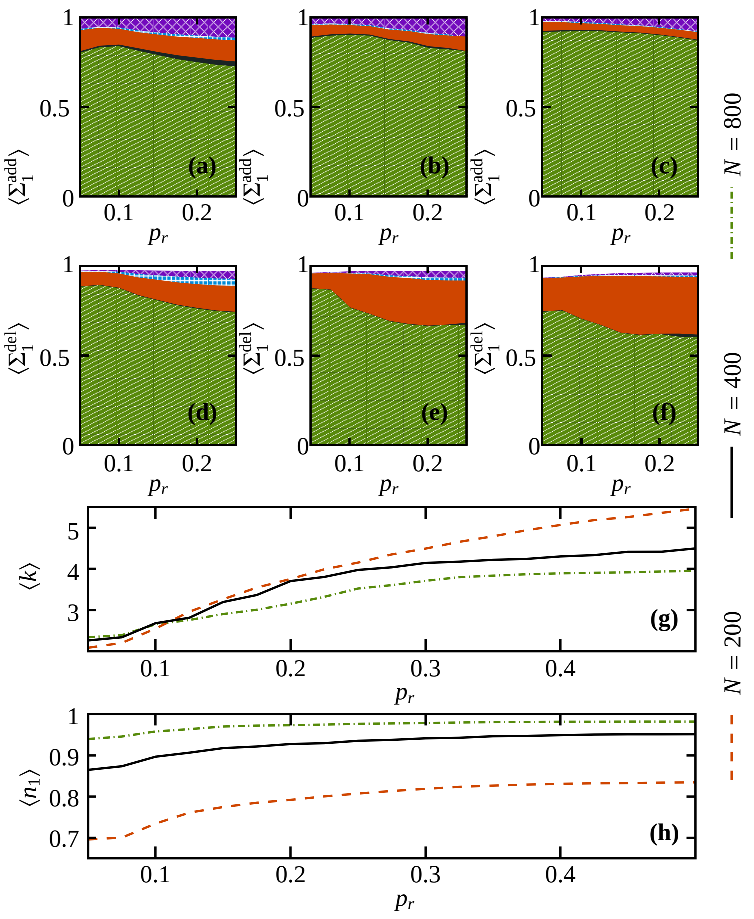

Figure 6.8: The system's dependence on the topological noise level (via the fraction of random rewirings $p_r$) for different system sizes $n$. Panels (a), (b) and (c) show the fraction $\langle\Sigma_1^{\mathrm{add}}\rangle$ of leading addition actions $\sigma_1^{\mathrm{add}}$ for systems of $n = 200$, 400 and 800. Panels (d), (e) and (f) show the fraction of preferred deletion actions for the same three system sizes, while (g) shows the average degree and (h) the average size of the largest connected component $\langle n_1\rangle$.

connected.

## 6.4 Discussion

Inspired by situations in the real world—particularly in the areas of diplomacy and politics—where individuals often aspire to a situation of profit or advantage, by balancing competing interests, we have presented, in this chapter, a general game-theoretic network problem—The diplomat's dilemma: How can an agent in a network simultaneously maximize closeness centrality and minimize degree? The diplomat's dilemma is one of the simplest models of such situations in a networked system, because, unlike in most models of a similar nature, the performance of agents solely depends on their position relative to others in the network, and not on any artificially ascribed extrinsic traits.

In order to run the objectives of the game, we have devised an iterative simulation where at every time step, an agent can delete its connection to a neighbor and add an edge to a second neighbor, based on the information it possesses about the network characteristics of vertices within its local neighborhood (upto second neighbors). The agents use strategies that they update by imitating the best performing neighbor within this information horizon. The dynamics are driven by occasional random moves and random permutations of the vectors encoding the strategies of the agents.

Despite the simplicity of our model, the time evolution of the simulation is strikingly complex, with quasi-stable states, trends, spikes and cascades of strategies among the agents. This complex dynamics is also captured in various metrics measuring different levels of network structure. Furthermore, the network structure and the agents' strategies directly influence one another. For example, If the agents stop deleting edges, the average degree of the network will grow rapidly, which is counter-productive to the objectives of the game. This feedback between the network topology, and the dynamical processes running on it is a central theme in the field of adaptive, coevolutionary networks [72]. We believe that all forms of social optimization involve such feedback loops, which is a strong motivation for studying adaptive networks.

The complexity of the time-evolution, especially in the network structural dynamics is more striking for intermediate system sizes. Indeed, many interesting features of our simulation are not emergent in the large-system limit, but rather present only for small sizes. Models in theoretical physics have traditionally focused on properties of the system as $n \to \infty$. In models of social systems however, extrapolating to infinite size is not necessarily a natural limit in the same way (it will of course be interesting to examine the limiting behavior of such models). We believe this is a good example of the dangers of taking the large-size limit by routine—the most interesting relevant features of the model may be neglected.

In a majority of the cases in our simulations, most of the agents use a strategy where they both delete, and attach to vertices according to the MAXC action. This implies that the agent first deletes the edge to the most central vertex in the second neighborhood (in the sense of a modified closeness centrality), and then reattaches to the most central vertex two steps away (before the deletion). In practice this means that an agent typically transfers an edge from its most central immediate neighbor to its most central neighbor two steps away. This strategy makes the agent move towards the center without increasing its degree, which clearly seems like a reasonable procedure in the context of the model. However, this strategy is not evolutionary stable in the presence of noise (hence the complex time evolution). The strategy also leads to networks with low levels of transitivity, i.e., there are a comparatively small number of triangles. Since forming a triangle introduces an extra edge, which is expensive, without changing the size of connected component, it seems reasonable that agents are reluctant to form such connections *per se* in our formulation of the problem.

Different strategies have different ability to invade one another. To test this we measure the deviation from random transitions from one dominating action to another (given the frequency of particular strategies), concluding for example that it is about twice as easy for RND to invade NO as a leading deletion action. Another interesting aspect is that (for some noise levels), as the system size increases, the network becomes both more connected (the relative fraction of vertices in the largest connected component increases), and more sparse (the average degree decreases). This is in sharp contrast to most other generative network models, but definitely consistent with the objectives of the general problem (where large connected components and low degrees are desired).

What does this result tell us about the real professional life of diplomats? Maybe that they can, by selfishly optimizing their positions in the network, self-organize to a connected business network where they need only a few business contacts, without knowing more about the network than the second neighborhood. However to make a stronger and more conclusive statement about the optimal strategy, more results are needed. This is something we hope to gather from future studies.

For the sake of flexibility, the definition of the problem as stated in this chapter, is deliberately vague. To turn it into a mathematically well-defined problem, one has to specify how the agents can affect their position in the network and what information they can use for this objective. There are of course many choices for how to do this. Although we believe our formulation is natural, it would be very interesting to rephrase these assumptions. A future enhancement would be to equip the agents with methods from the machine learning community to optimize their position, and to tune the amount of information accessible to the agents. Following the prescription of Jackson and Wolinsky [83] one could add an additional constraint on the model that would require an edge to represent an agreement between both vertices, so that an agent $i$ cannot add an edge $(i, j)$ unless both $i$ and $j$ find this mutually profitable—so-called pairwise stability. It would then be interesting to see if we still retain the complex evolution of the network, as this additional constraint reduces the number of possible graphs than can be generated by our game.

One of the main drawbacks with this type of mechanistic modeling of social information processes [76, 66, 131, 132], is that they are very hard to validate. Information spreading in social systems is neither routed from agent to agent like the information packets in the Internet, nor do they spread in the same fashion as epidemics. Instead the spreading dynamics is (usually) content dependent. Different types of information may be spreading over different social networks, following different dynamic rules. There are some promising datasets for studying social information spreading. For example, networks of blogs, Internet communities, or social networking sites generate large amounts of potentially valuable data, although these data sets are not necessarily conducive to the questions that adaptive models such as the one described in this chapter seek to address. In the near future, we hope there will be sufficient enhancements in devising such models, so that we can make extensive comparisons with data from the real world.

CHAPTER VII

# Conclusion

Broadly speaking, the study of networks in the existing scientific literature can be separated into three areas. In the first case, a lot of work has been done on defining and measuring structural properties of networks, such as, degree distributions, clustering coefficients, assortativity, among others. In the second approach, researchers have used networks as templates on which to simulate various dynamical processes such as epidemic spreading, voter opinion, phase synchronization, biological processes and so on. In addition to this, they have studied the structural evolution of networks under various sets of rules, to try and explain the origin of its properties. Finally, there is a relatively new line of enquiry which seeks to combine both aspects, in that there is a direct link between the structure as well as function, with one directly dependent on the other, and thus the properties of both have to be treated simultaneously. In this dissertation, using methods and ideas drawn from statistical physics, computer science and discrete mathematics, we have investigated the properties of networks using all three approaches.

In Chapter II, we have proposed an alternative and perhaps more realistic measure of network robustness, the presence (or not) of bicomponents—sets of vertices in which any two in the set are connected by at least two independent paths. Using standard network models, we have shown that for a given network, there are no small bicomponents, however there exists at most one giant bicomponent that is nested within the regular giant component. In addition we have also provided expressions for the expected size of the giant bicomponents along with their variation as a function of vertex removal. We have shown that as vertices are removed, the giant bicomponent persists down to the point at which the ordinary giant component disappears, but with an unusual third-order transition at that point that ensures that the size of the bicomponent will be small as we approach the transition. We have tested our theoretical results on a series of data taken from networks in the real world. The real-world networks investigated are found to be quite robust in the sense of having large giant bicomponents and moreover the existence of these bicomponents is, at least in some cases, itself robust to the deletion of vertices. In practice, however, although the giant bicomponent may persist as vertices are removed from the network, its size dwindles rapidly so that large portions of the network lose robust connection considerably before the transition point at which the giant bicomponent finally vanishes. In this respect, it seems that real world networks (with some notable exceptions) are remarkably similar to exactly solvable random graph models. An obvious generalization of this problem is to extend it to the case of directed networks. An immediate challenge then would be to define analogs for weak and strong connections (as in the case of the regular 1-components) for bicomponents and other higher components. In addition to this, one would have to devise an appropriate algorithm to find these types of directed bicomponents in real networks. It is not obvious that the results would be the same as in the undirected case, and it certainly merits further enquiry.

In Chapter III, we have proposed and studied in detail a model of random tripartite hypergraphs. Our study has been motivated by the emergence of new types of social networks, such as folksonomies, a tripartite structure consisting of users that apply a short text description called tags, to a set of resources. In particular we have defined the tripartite analog of basic network measures, such as degree distributions, the various types of components, and projections onto individual vertex types, and using an extension of the configuration model, calculated a variety of statistical properties in the limit of large network size. Among other things we have calculated the explicit degree distributions for projected networks, conditions for the emergence of a giant component, the size of the giant component when there is one, and the location of the percolation threshold for site percolation on the network. We have compared our results against measurements of computer-generated random hypergraphs and a real-world tripartite network, the folksonomy of the online photo-sharing web site Flickr. In the latter case, we have focused on the degree distributions of projections of the hypergraph onto one vertex type and find that in some instances the theory makes predictions in moderately good agreement with the observations while in others the agreement is poorer. We have shown, however, that the agreement between the theory and observation is much improved, if we remove instances of multiple tagging—instances in which a user applies many tags to the same photo or the same tag to many photos. The results seems to suggest that, at least in terms of the connection structure as measured by the degree distribution, the primary disagreement between model and theory is due to trivial loop structures, rather than any subtle social effects—although this warrants further investigation. In a follow-up paper [153], we extended our model by defining hypergraphs analogs of the clustering coefficient, assortativity, degree correlations among other topological measures and then measured them on real world tripartite networks. Based on our findings, a recent paper by Bradde and Bianconi [26] extended our theoretical analysis by including correlations between the different classes of vertices.

While these two chapters have dealt with properties of "static networks", Chapters IV and V investigate the properties of networks whose structure evolves in time. In Chapter IV we have studied models of the time evolution of networks in which, in addition to the widely considered case of addition of vertices, we also include vertex removal. Using a master-equation approach and with the aid of generating functions, we have given exact solutions for cases in which vertices are added and removed at the same rate, and cases in which the rate of addition exceeds the rate of removal, which we regard as a simple model for the growth of, for example, the World Wide Web. Among the most interesting of our results is where newly appearing vertices attach to others using a linear preferential attachment mechanism, whereby vertices gain new edges in proportion to the number they already possess. If the rate of adding vertices is balanced with that of losing them, then we find that the network has degrees distributed according to a stretched exponential. If however the rate of adding vertices exceeds that of removing them—the regime of net growth—the degree distribution follows a power law, with an exponent $\gamma$ dependent on the rate and that assumes values in the range $3 \leq \gamma < \infty$, diverging as the growth rate tends to zero. This is of interest for a number of reasons. First, it shows that power-law behavior can be rigorously established in networks that grow but also lose vertices. Most previous analytic models of network growth have focused solely on vertex addition (as discussed in detail in Sec 1.6.1). The results presented here demonstrate that the widely studied mechanism of preferential attachment for generating power-law behavior also works in this regime. An interesting implication of our model is that the origin of power-laws in real networks might very well be a consequence of robust growth. The Web, for example, has certainly being enjoying a period of very vigorous growth since its appearance in the early 1990s, but it could be that this is a sign primarily of its youth, and that as the network matures its size will grow more slowly, the vertices added being more nearly balanced by those taken away. Were this to happen, we would expect to see the exponent of the degree distribution grow larger. We note that a number of authors have extended our model by incorporating more complex modes of growth—see for example [81, 134, 20]. An interesting application of the model was proposed by Karrer and Ghoshal [85] where they used the mathematical framework to show that degree distributions of networks can be dynamically preserved from disruptions, by re-attaching nodes and edges using appropriate rules to compensate for various types of attacks.

In Chapter V we have used the theoretical insight afforded by the previous chapter in a practical application. We considered the case of distributed networks, whose structure can be manipulated by appropriate adjustments of the rules by which vertices join and leave the network. We have shown that, with some minor constraints, it is possible to devise appropriate attachment rules for vertices, to generate networks with any degree distributions that we desire. We have also described in detail a local algorithm based on biased random walks via which the the attachment schemes can be realized in practice. As an example application, we have employed our ideas in the construction of a peer-to-peer file sharing network, with a topology optimized to minimize the time it takes to search files, as well as the average bandwidth requirements.

Finally, in Chapter VI, we have devised and studied a model where the structural and dynamical evolution of the network are intricately linked—one is the direct result of the other and vice versa. In our model, a set of agents compete with each other to optimize their position in the network (as measured by their closeness centrality), while attempting to keep the costs of doing so (their degree) low. In other words the success of an agent increases with its closeness centrality, while it decreases with its degree. Such a situation may occur in diplomacy, lobbying or business networks, where an agent wants to be central in the network (for the purpose of having access to information and thus be more actively involved in influencing outcomes) but not at the expense of having too many direct contacts (which takes up too much time and effort). The dynamics proceed by the agents deleting edges and attaching new edges to neighbors two steps away, according to strategies based on local information. Once in a while the agents evaluate the strategies of the neighborhood and imitate the best performing neighbor to optimize their strategy. As the vertices of our model have no additional traits, their competitive situation is completely determined by their position in the network—the time evolution of strategies is immediately tied to the evolution of network structure. We have shown with the aid of extensive computer simulations, that the evolutionary trajectories are strikingly complex having long periods of relative stability followed by sudden transitions, spikes, or chaotic periods visible in both the strategies and the network structure. One such instability is manifested in a transient fragmentation of the network. We have demonstrated that as the size of the network is increased, this instability vanishes and the network has larger connected components, accompanied with a decreasing fraction of links—thus, with a growing number of actors the system gets better at achieving the common goal of being connected and keeping the degree low. Among other things, our simulations have revealed that the network dynamics never reaches a fixed point of passivity (where the network is largely static), this suggests situation similar to the Red Queen hypothesis—agents have to keep on networking to maintain their success.

While the work presented in this dissertation, is fairly broad in scope and investigates the properties of networks from a variety of angles, this represents merely "a drop in the ocean" compared to the sheer scale of active research in the field, as well as the potential future questions (of which there are many). Nevertheless, we hope that the material and the ideas presented here will do some justice in advancing our knowledge of the field, and we look forward to many future developments building on these ideas.

# BIBLIOGRAPHY

# BIBLIOGRAPHY

[1] M. Abramowitz and I.A. Stegun (eds.), *Handbook of mathematical functions*, Dover Publishing, New York, 1974.

[2] L.A. Adamic and B.A. Huberman, *Power-law distributions of the world wide web*, Science **287** (2001), 2115a.

[3] L.A. Adamic, R.M. Lukose, A.R. Puniyani, and B.A. Huberman, *Search in power-law networks*, Phys. Rev. E **64** (2001), 046135.

[4] W. Aiello, F. Chung, and L. Lu, *A random graph model for massive graphs*, Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (New York), Association of Computing Machinery, 2000, p. 171.

[5] R. Albert and A.-L. Barabási, *Topology of evolving networks: Local events and universality*, Phys. Rev. Lett. **85** (2000), 5234.

[6] ———, *Statistical mechanics of complex networks*, Rev. Mod. Phys. **74** (2002), 47.

[7] R. Albert, H. Jeong, and A.-L. Barabási, *Diameter of the world-wide web*, Nature **401** (1999), 130.

[8] ———, *Attack and error tolerance of complex networks*, Nature **406** (2000), 378.

[9] B. Bollobás, *A probabilistic proof of an asymptotic formula for the number of labeled regular graphs*, European J. Combin. **1** (1980), 311.

[10] B. Bollobás (ed.), *Random graphs*, Academic Press, New York, 1985.

[11] B. Bollobás and O. Riordan, *The diameter of a scale free random graph*, Combinatorica **24** (2004), 5.

[12] B. Bollobás, O. Riordan, J. Spencer, and G. Tusnády, *The degree sequence of a scale-free random graph process*, Random Struct. Algorithms **18** (2001), 279.

[13] A.-L. Barabási and R. Albert, *Emergence of scaling in random networks*, Science **286** (1999), 509.

[14] R.J. Aumann, *Game theory*, The New Palgrave Dictionary of Economics (New York) (S.N. Durlauf and L.E. Blume, eds.), Palgrave Macmillan, 2008, p. 460.

[15] R. Axelrod (ed.), *The evolution of cooperation*, Basic Books, New York, 1984.

[16] R. Axelrod (ed.), *The complexity of cooperation: Agent-based models of competition and collaboration*, Princeton University Press, Princeton, 1997.

[17] A. Barrat and M. Weigt, *On the properties of small-world network models*, Eur. Phys. J. B **13** (2000), 547.

[18] J. Barron, *Power surge blacks out northeast*, New York Times, August 2003.

[19] P.S. Bearman, J. Moody, and K. Stovel, *Chains of affection: The structure of adolescent romantic and sexual networks*, Am. J. Sociol. **110** (2004), 44.

[20] E. Ben-Naim and P.L. Krapivsky, *Addition-deletion networks*, J. Phys. A: Math. Theor. **40** (2007), 8607.

[21] E.A. Bender and E.R. Canfield, *The asymptotic number of labeled graphs with given degree sequences*, J. Combin. Theory Ser. A **24** (1978), 296.

[22] G. Bianconi and A.-L. Barabási, *Bose-einstein condensation in complex networks*, Phys. Rev. Lett. **90** (2003), 078701.

[23] G. Bianconi and A. Capocci, *Number of loops of size h in growing scale-free networks*, Phys. Rev. Lett. **90** (2003), 078701.

[24] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.U. Hwang, *Complex networks: Structure and dynamics*, Phys. Rep. **424** (2006), 175.

[25] P.F. Bonacich, *Power and centrality: A family of measures*, Amer. J. Sociol. **92** (1989), 1170.

[26] S. Bradde and G. Bianconi, *Percolation transition in correlated hypergraphs*, arXiv:0905.4455v1, 2009.

[27] S. Brin and L. Page, *The anatomy of a large-scale hypertextual Web search engine*, Comput. Netw. **30** (1998), 107.

[28] S.R. Broadbent and J.M. Hammersley, *Percolation processes: I. crystals and mazes*, Proc. Cambridge Philos. Soc. **53** (1957), 629.

[29] A. Broder and A. Karlin, *Bounds on the cover time*, J. Theor. Probab. **2** (1989), 101.

[30] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener, *Graph structure in the web*, Comput. Netw. **33** (2000), 309.

[31] Z. Burda, J. Jurkiewicz, and A. Krzywicki, *Network transitivity and matrix models*, Phys. Rev. E **69** (2004), 026106.

[32] G. Caldarelli (ed.), *Scale-free networks*, Oxford University Press, Oxford, 2007.

[33] G. Caldarelli, R. Pastor-Satorras, and A. Vespignani, *Structure of cycles and local ordering in complex networks*, Eur. Phys. J. B **38** (2004), 183.

[34] D.S. Callaway, M.E.J. Newman, S.H. Strogatz, and D.J. Watts, *Network robustness and fragility: Percolation on random graphs*, Phys. Rev. Lett. **85** (2000), 5468.

[35] C. Cattuto, V. Loreto, and L. Pietronero, *Semiotic dynamics and collaborative tagging*, Proc. Nat. Acad. Sci. USA **104** (2007), 1461.

[36] C. Cattuto, C. Schmitz, A. Baldassarri, V.D.P. Servedio, V. Loreto, A. Hotho, M. Grahl, and G. Stumme, *Network properties of folksonomies*, AI Commun. **20** (2007), 245.

[37] F. Chung and R. Graham, *Quasi random hypergraphs*, Proc. Nat. Acad. Sci. USA **86** (1989), 8175.

[38] F. Chung and L. Lu, *The average distances in random graphs with given expected degrees*, Proc. Natl. Acad. Sci. USA **99** (2002), 15879.

[39] ______, *Connected components in random graphs with given degree sequences*, Ann. Comb. **6** (2002), 1125.

[40] ______, *Coupling online and offline analyses for random power law graphs*, Internet Mathematics **1** (2004), 409.

[41] R. Cohen, K. Erez, D. ben Avraham, and S. Havlin, *Resilience of the internet to random breakdowns*, Phys. Rev. Lett. **85** (2000), 4626.

[42] C. Cooper, *The cores of random hypergraphs with a given degree sequence*, Rand. Struct. Algorithms **25** (2004), 353.

[43] C. Cooper, A. Frieze, and J. Vera, *Random deletion in a scale-free random graph process*, Internet Mathematics **1** (2004), 463.

[44] A. Davis, B.B. Gardner, and M.R. Gardner (eds.), *Deep South*, University of Chicago Press, Chicago, 1941.

[45] N.G. de Bruijn (ed.), *Asymptotic methods in analysis*, Dover, New York, 1981.

[46] P.G. de Gennes, *La notion de percolation*, La Recherche **72** (1976), 919.

[47] A. Dembo and A. Montanari, *Finite size scaling for the core of large random hypergraphs*, Ann. Appl. Probab. **18** (2008), 5.

[48] D.J. de S. Price, *Networks of scientific papers*, Science **149** (1965), 510.

[49] ______, *A general theory of bibliometric and other cumulative advantage processes*, J. Amer. Soc. Inform. Sci. **27** (1976), 292.

[50] S.N. Dorogovtsev and J.F.F. Mendes, *Scaling behaviour of developing and decaying networks*, Europhys. Lett. **52** (2000), 33.

[51] ______, *Effect of the accelerating growth of communication networks on their structure*, Phys. Rev. E. **63** (2001), 025101.

[52] ______, *Evolution of networks*, Adv. Phys. **51** (2002), 1079.

[53] S.N. Dorogovtsev and J.F.F. Mendes (eds.), *Evolution of networks: From biological nets to the internet and www*, Oxford University Press, Oxford, 2003.

[54] S.N. Dorogovtsev, J.F.F. Mendes, and A.N. Samukhin, *Structure of growing networks with preferential linking*, Phys. Rev. Lett. **85** (2000), 4633.

[55] J. Duch and A. Arenas, *Community detection in complex networks using extremal optimization*, Phys. Rev. E **72** (2005), 027104.

[56] P. Erdős and A. Rényi, *On random graphs*, Publ. Math. Debrecen **6** (1959), 290.

[57] ______, *On the evolution of random graphs*, Publ. Math. Inst. Hung. Acad. Sci. **5** (1960), 17.

[58] ______, *On the strength of connectedness of a random graph*, Acta. Math. Acad. Sci. Hungar. **12** (1961), 261.

[59] P. Erdős and L. Lovász, *Problems and results on 3-chromatic hypergraphs and some related questions*, Colloq. Math. Soc. Janos Bolyai **10** (1975), 609.

[60] M. Faloutsos, P. Faloutsos, and C. Faloutsos, *On power-law relationships of the internet topology*, Computer Communications Rev. **29** (1999), 251.

[61] L.C. Freeman, *A set of measures of centrality based on betweenness*, Sociometry **40** (1977), 35.

[62] A. Fronczak, J.A. Holyst, M. Jedynak, and J. Sienkiewiecz, *Higher order clustering coefficients in Barabasi-Albert networks*, Physica A **316** (2002), 688.

[63] G. Ghoshal, *Some New Applications of Network Growth Models*, Dynamics On and Off Complex Networks: Applications to Biology Computer Science and the Social Sciences (N. Ganguly, A. Deutsch, and A. Mukherjee, eds.), Birkhäuser, Boston, 2009, p. 217.

[64] G. Ghoshal and M.E.J. Newman, *Growing distributed networks with arbitrary degree distributions*, Eur. Phys. J. B **58** (2007), 175.

[65] G. Ghoshal, V. Zlatić, G. Caldarelli, and M.E.J. Newman, *Random hypergraphs and their applications*, Phys. Rev. E **79** (2009), 066118.

[66] S. Gil and D.H. Zanette, *Coevolution of agents and networks: Opinion spreading and community disconnection*, Phys. Lett. A **356** (2006), 89.

[67] C. Gkantsidis, M. Mihail, and A. Saberi, *Random walks in peer-to-peer networks*, Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (New York), Institute of Electrical and Electronics Engineers, 2004.

[68] C. Godsil and G. Royle (eds.), *Algebraic graph theory*, Springer-Verlag, New York, 2001.

[69] O. Gorlitz, S. Sizov, and S. Staab, *Tagster -tagging-based distributed content sharing*, The Semantic Web (S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarikis, eds.), vol. 5021, Springer-Verlag, Berlin, 2008, p. 807.

[70] P. Grassberger, *On the critical behavior of the general epidemic process and dynamic percolation*, Math. Biosci. **63** (1983), 57.

[71] A. Grönlund, K. Sneppen, and P. Minnhagen, *Correlations in networks associated to preferential growth*, Phys. Scr. **71** (2005), 680.

[72] T. Gross and B. Blausius, *Adaptive coevolutionary networks: a review*, J. Roy. Soc. Interface **5** (2008), 259.

[73] J.M. Hammersley, *Percolation processes: II. the connective constant*, Proc. Cambridge Philos. Soc. **53** (1957), 642.

[74] P.W. Holland and S. Leinhardt., *Some evidence on the transitivity of positive interpersonal sentiment*, Am. J. Sociol. **72** (1972), 1205.

[75] ______, *An exponential family of probability distributions for directed graphs*, J. Amer. Statist. Assoc. **76** (1981), 33.

[76] P. Holme and G. Ghoshal, *Dynamics of networking agents competing for high centrality and low degree*, Phys. Rev. Lett. **96** (2006), 098701.

[77] ______, *The diplomats dilemma: Maximal power for minimal effort in social networks*, Adaptive Networks: Theory, Models and Applications (T. Gross and H. Sayama, eds.), Springer-Verlag, Heidelberg, 2009.

[78] P. Holme and J. Zhao, *Exploring the assortativity-clustering space of a network's degree sequence*, Phys. Rev. E **75** (2007), 046111.

[79] T. Hong, *Performance*, Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology (Sebastopol, CA) (A. Oram, ed.), O'Reilly and Associates, 2001, p. 203.

[80] J.E. Hopcroft and R.E. Tarjan, *Efficient algorithms for graph manipulation*, Comm. ACM **16** (1973), 372.

[81] K. Iguchi and H.S. Yamda, *General connectivity distribution functions for growing networks with preferential attachment of fractional power*, J. Math. Phys. **48** (2007), 113303.

[82] M. Jackson (ed.), *Social and Economic Networks*, Princeton University Press, Princeton, 2008.

[83] M.O. Jackson and A. Wolinsky, *A Strategic Model of Social and Economic Networks*, J. Econ. Theory **71** (1996), 44.

[84] D. Kahneman, *Maps of bounded rationality: psychology for behavioral economics*, The American Economic Review **93** (2003), 1449.

[85] B. Karrer and G. Ghoshal, *Preservation of network degree distributions from non-uniform failures*, Eur. Phys. J. B **62** (2008), 239.

[86] L. Katz, *A new status index derived from sociometric analysis*, Psychometrika **18** (1953), 39.

[87] S.A. Kauffman, *Metabolic stability and epigenesis in randomly connected nets*, J. Theor. Bio. **22** (1969), 437.

[88] ______, *Gene regulation networks: A theory for their structure and global behavior*, Current topics in Developmental Biology (A. Moscana and A. Monroy, eds.), vol. 6, Academic Press, New York, 1971, p. 145.

[89] R. Kinney, P. Crucitti, R. Albert, and V. Latora, *Modelling cascading failures in the North American power grid*, Eur. Phys. J. B **46** (2005), 101.

[90] S. Kirkpatrick, *Percolation and conduction*, Rev. Mod. Phys. **45** (1973), 574.

[91] J.M. Kleinberg, S.R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, *The Web as a graph: Measurements, models and methods*, Proceedings of the 5th Annual International Conference on Combinatorics and Computing (Berlin) (T. Asano, H.I mai, D.T. Lee, S.-I. Nakano, and T. Tokuyama, eds.), Lecture Notes in Computer Science, vol. 1627, Springer-Verlag, 1999, p. 1.

[92] D. Knoke. (ed.), *Political networks: The structural perspective*, Cambridge University Press, Cambridge, 1990.

[93] P.L. Krapivsky and S. Redner, *Organization of growing random networks*, Phys. Rev. E **63** (2001), 066123.

[94] ______, *Finiteness and fluctuations in growing networks*, J. Phys. A **35** (2002), 9517.

[95] ______, *A statistical physics perspective on web growth*, Comput. Netw. **39** (2006), 261.

[96] P.L. Krapivsky, S. Redner, and F. Leyvraz, *Connectivity of growing random networks*, Phys. Rev. Lett. **85** (2000), 4629.

[97] R. Lambiotte and M. Ausloos, *Collaborative Tagging as a Tripartite network*, Proceeding of the 6th International Conference on Computer Science, Part III, Lecture Notes in Computer Science, vol. 3993, Springer-Verlag, Berlin, 2006, p. 1114.

[98] J. Leskovec, J. Kleinberg, and C. Faloutsos, *Graph Evolution: Densification and Shrinking Diameters*, ACM Transactions on Knowledge Discovery from Data **1** (2007), 1.

[99] K. Lindgren and M. G. Nordahl, *Evolutionary dynamics of spatial games*, Physica D **75** (1994), 292.

[100] A.J. Lotka, *The frequency distribution of scientific productivity*, Journal of the Washington Academy of Sciences **16** (1926), 317.

[101] T. Luczak, *Sparse random graphs with a given degree sequence*, Proceedings of the Symposium on Random Graphs, Poznań (New York) (A.M. Frieze and T. Luczak, eds.), John-Wiley, 1992, p. 165.

[102] M. Molloy and B. Reed, *The critical point for random graphs with a given degree sequence*, Random Struct. Algorithms **6** (1995), 161.

[103] C. Moore, G. Ghoshal, and M.E.J. Newman, *Exact solutions for models of evolving networks with addition and deletion of nodes*, Phys. Rev. E **74** (2006), 036121.

[104] C. Moore and M.E.J. Newman, *Exact solution of site and bond percolation on small-world networks*, Phys. Rev. E **62** (2000), 7059.

[105] J.L. Moreno (ed.), *Who shall survive?*, Beacon House, Beacon, New York, 1934.

[106] M.E.J. Newman, *The structure of scientific collaboration networks*, Proc. Natl. Acad. Sci. USA **98** (2001), 404.

[107] ______, *Assortative mixing in networks*, Phys. Rev. Lett. **89** (2002), 208701.

[108] ______, *The structure and function of complex networks*, SIAM Review **45** (2003), 167.

[109] ______, *Power laws, Pareto distributions and Zipf's law*, Contemp. Phys. **46** (2005), 323.

[110] ______, *Finding community structure in networks using the eigenvectors of matrices*, Phys. Rev. E **74** (2006), 036104.

[111] ______, *Component sizes in networks with arbitrary degree distributions*, Phys. Rev. E **76** (2007), 045101.

[112] ______, *Random graphs with clustering*, arXiv:0903.4009v1, 2009.

[113] M.E.J. Newman, A.-L. Barabási, and D.J. Watts (eds.), *The structure and dynamics of networks*, Princeton University Press, Princeton, 2006.

[114] M.E.J. Newman and G.T. Barkema (eds.), *Monte carlo methods in statistical physics*, Oxford University Press, Oxford, 1999.

[115] M.E.J. Newman, S. Forrest, and J. Balthrop, *Email networks and the spread of computer viruses*, Phys. Rev. E **66** (2002), 035101.

[116] M.E.J. Newman and G. Ghoshal, *Bicomponents and robustness of networks to failures*, Phys. Rev. Lett. **100** (2008), 138701.

[117] M.E.J. Newman, S.H. Strogatz, and D.J. Watts, *Random graphs with arbitrary degree distributions and their applications*, Phys. Rev. E **64** (2001), 026118.

[118] M.E.J Newman and D.J. Watts, *Scaling and percolation in the small-world network*, Phys. Rev. E **60** (1999), 7332.

[119] M.E.J Newman and R.M. Ziff, *Efficient monte carlo algorithm and high-precision results for percolation*, Phys. Rev. Lett. **85** (2000), 4104.

[120] M. Nowak and R. M. May, *Evolutionary games and spatial chaos*, Nature **359** (1992), 826.

[121] M. Nowak and K. Sigmund, *A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoner's dilemma game*, Nature **346** (1992), 56.

[122] G. Palla, I. J. Farkas, P. Pollner, I. Derényi, and T. Vicsek, *Fundamental statistical features and self-similar properties of tagged networks*, New J. Phys. **10** (2008), 123026.

[123] J. Park and M.E.J. Newman, *The statistical mechanics of networks*, Phys. Rev. E **70** (2004), 066117.

[124] G. Paul, T. Tanizawa, S. Havlin, and H.E. Stanley, *Optimization of robustness of complex networks*, Eur. Phys. J. B **38** (2004), 187.

[125] J.J Potterat, L. Phillips-Plummer, S.Q. Muth, R.B. Rothenburg, D.E. Woodhouse, T.S. Maldonado-Long, H.P. Zimmerman, and J.B. Muth, *Risk network structure in the early epidemic phase of HIV transmission in Colorado Springs*, Sexually Transmitted Infections **78** (2002), i159.

[126] S. Redner, *How popular is your paper? an empirical study of the citation distribution*, Eur. Phys. J. B **4** (1998), 131.

[127] M. Ripenau, I. Foster, and A. Iamnitchi, *Mapping the Gnutella network: Properties of large-scale peer-to-peer systems and implications for system design*, IEEE Internet Computing **6** (2002), 50.

[128] J.P. Ritter, *Why Gnutella can't scale. No, really*, http://www.darkridge.com/ jpr5/doc/gnutella.html, 2000.

[129] J.L. Rodgers and W.A. Nicewander, *Thirteen ways to look at the correlation coefficient*, The American Statistician **42** (1988), 59.

[130] F.J. Roethlisberger and W.J. Dickson (eds.), *Management and the Worker*, Harvard University Press, Cambridge, MA, 1939.

[131] M. Rosvall and K. Sneppen, *Modelling dynamics of information networks*, Phys. Rev. Lett. **91** (2003), 178701.

[132] ______, *Modeling self-organization of communication and topology in social networks*, Phys. Rev. E **74** (2006), 016108.

[133] G. Sabidussi, *The centrality index of a graph*, Psychometrika **31** (1966), 581.

[134] J. Saldaña, *Continuum formalism for modeling growing networks with deletion of nodes*, Phys. Rev. E **75** (2007), 027102.

[135] N. Sarshar, P.O. Boykin, and V.P. Roychowdhury, *Scaleable percolation search in power-law networks*, Proceedings of the 4th International Conference on Peer-to-Peer Computing (New York), IEEE Computer Society, 2004, p. 2.

[136] N. Sarshar and V.P. Roychowdhury, *Scale-free and stable structures in complex ad hoc networks*, Phys. Rev. E **69** (2004), 026101.

[137] J. Schmidt-Pruzan and E. Shamir, *A threshold for perfect matching in d-pure random hypergraphs*, Discrete Math. **43** (1982), 315.

[138] ______, *Component structure in the evolution of random hypergraphs*, Combinatorica **5** (1985), 81.

[139] R. Solomonoff and A. Rapoport, *Connectivity of random nets*, Bull. Math. Biophys. **13** (1951), 107.

[140] D. Stauffer and A. Aharony (eds.), *Introduction to Percolation Theory*, Taylor and Francis, London, 1994.

[141] K. Stephenson and M. Zelen, *Rethinking centrality: Methods and applications*, Soc. Netw. **11** (1989), 1.

[142] D. Strauss, *On a general class of models for interaction*, SIAM Review **28** (1986), 513.

[143] B. Tadić, *Temporal fractal structures: Origin of power laws in the world-wide web*, Physica A **314** (2002), 278.

[144] H. Timmons, *2 communication cables in the mediterranean are cut*, New York Times, January 2008.

[145] L.M. van Valen, *A new evolutionary law*, Evolutionary Theory **1** (1973), 1.

[146] J. von Neumann and O. Morgenstern (eds.), *The theory of games and economic behavior*, Princeton University Press, Princeton, 1944.

[147] S. Wasserman and K. Faust (eds.), *Social Network Analysis: Methods and Applications*, Cambridge University Press, Cambridge, 1994.

[148] D.J. Watts (ed.), *Six degrees: The science of a Connected Age*, Norton, New York, 2003.

[149] D.J. Watts and S.H. Strogatz, *Collective dynamics of 'small-world' networks*, Nature **393** (1998), 440.

[150] M. Weber (ed.), *The theory of social and economic organization*, Oxford University Press, New York, 1947.

[151] D.B. West (ed.), *Introduction to graph theory*, Prentice Hall, Upper Saddle River, NJ, 1996.

[152] J. Westbrook and R.E. Tarjan, *Maintaining bridge-connected and biconnected components on-line*, Algorithmica **7** (1992), 433.

[153] V. Zlatić, G. Ghoshal, and G. Caldarelli, *Hypergraph topological quantities for tagged social networks*, arXiv:0905.0976v1, 2009.