Modeling and Designing Real–World Networks

Michael Kaufmann[1] and Katharina Zweig[2]

[1] University of Tübingen,
Wilhelm–Schickard–Institute für Informatik,
Sand 14, 72076 Tübingen, Germany
mk@informatik.uni-tuebingen.de
[2] Eötvös Loránd University (ELTE)
Department of Biological Physics,
Pazmany P. stny 1A,
1117 Budapest, Hungary
nina@angel.elte.hu

Abstract. In the last 10 years a new interest in so–called real–world graph structures has developed. Since the structure of a network is crucial for the processes on top of it a well–defined network model is needed for simulations and other kinds of experiments. Thus, given an observable network structure, models try to explain how they could have evolved. But sometimes also the opposite question is important: given a system with specific constraints what kind of rules will lead to a network with the specified structure? This overview article discusses first different real–world networks and their structures that have been analyzed in the last decade and models that explain how these structures can emerge. This chapter concentrates on those structures and models that are very simple and can likely be included into technical networks such as P2P-networks or sensor networks. In the second part we will then discuss how difficult it is to design local network generating rules that lead to a globally satisfying network structure.

1 Introduction

Since the 1950s a considerable part of graph theory was devoted to the study of theoretical graph models, so–called random graphs that are created by a random process. Especially the random graph models $G(n, m)$ defined by Erdös and Rényi in 1959 [19] and the $G(n, p)$-model introduced by Gilbert [22] proved themselves to be very handy and well analyzable [9]. Other simple graph classes like regular graphs, grid graphs, planar graphs, and hypercubes were also defined and analyzed, and many of their properties proved to be useful in algorithm design, making hard problems considerable easier. It is reasonable that most real–world networks neither belong to any of the simple graph classes nor that they can be fully modeled as random graphs. But on the other hand, it seemed to be reasonable that at least some real-world networks can be considered to be random on a global scale, for example the so–called weblink graph: the weblink graph represents webpages as vertices and connects two vertices with a (directed)

edge when the first page links to the second. Although it can be assumed that most of the time similar pages link to each other, the vast number of single decisions could be assumed to randomize this structure on a global scale. Thus, it came with a surprise when in 1999, Barabási and Albert showed that the real weblink graph cannot be modeled by a random graph since most of the webpages have only a low degree of in- and outcoming links while some have a huge degree, e.g., Yahoo or Google [6]. This phenomenon could not be explained by any of the classic random graph models. Another finding by Watts and Strogatz showed that real–world networks combine two properties that are not captured by any of the classic network models: real–world networks tend to be clustered, i.e., the neighbors of any vertex are likely to be connected, similar to grid graphs, but at the same time the average distance between all the vertices is much lower than expected in such a grid–like graph and resembles that of a random graph [48].

These findings opened a new field in between empirical sciences, mostly physics, biology, and the social sciences, and theoretical sciences as computer science and mathematics. This field is called *complex network science* [38, 5][3]. In a very broad definition, we will denote by *complex network science* all research that can be subsumed under the following three perspectives:

1. **Complex Network Analysis**: measures and algorithms introduced to understand the special structure of real–world networks by differentiating them from established graph models.
2. **Complex Network Models**: models that capture essential structural properties of real–world networks and algorithms that construct them.
3. **Processes on Complex Networks**: analysis of the outcome of a process or algorithm on a given network structure.

In the following we will summarize the progress in the field of complex network models and show its relevance for algorithm design and computer simulations. Section 2 gives the necessary definitions and Section 3 discusses two principally different ways of modeling real–world network data, the *data–driven* and the *mechanistic* approach. Section 4 focuses on different mechanistic network models for real–world networks. Section 5 discusses some of the difficulties in designing networks for a set of independent agents. A summary and discussion of open problems is given in Section 6. Section 7 gives a list of well-organized resources for further reading.

2 Definitions

A graph G is a pair of sets (V, E) where $V = \{v_1, \ldots, v_n\}$ denotes a set of vertices and $E \subseteq V \times V$ denotes a *relation* between these vertices. If all edges of a graph are given as unordered pairs of vertices, the graph is said to be *undirected*. The degree $deg(v)$ of vertex v is defined as the number of edges it is element of.

[3] Complex network science strongly overlaps with a new field called *web science*, introduced by (among others) Sir Berners-Lee [8].

We will denote undirected edges e by pairs of vertices in simple brackets (v, w). A *weight function* $\omega : E \to \mathbb{R}$ assigns a weight to each edge. If the graph is unweighted, it is convenient to set $\omega(e) := 1$ for all edges. A *path* $P(s,t)$ from vertex s to vertex t is an ordered set of consecutive edges $\{e_1, e_2, \ldots, e_k\} \subseteq E$ with $e_1 = (s, v_1), e_k = (v_{k-1}, t)$ and $e_i = (v_{i-1}, v_i)$, for all $1 < i < k$. The *length of a path* $l(P(s,t))$ is defined as the sum over the weights of the edges in the path:

$$l(P(s,t)) = \sum_{e \in P(s,t)} \omega(e). \tag{1}$$

A path $P(s,t)$ is a *shortest path* between s and t if it has minimal length of all possible paths between s and t. The *distance* $d(s,t)$ between s and t is defined as the length of a shortest path between them. If there is no path between any two vertices, their distance is ∞ by definition. The *diameter* of a given graph is defined as the maximal distance between any two of its vertices if the graph is connected and defined to be ∞ if it is unconnected.

We will denote as a *real–world network* or, synonymously, as a *complex network* any network that presents an abstract view on a complex system, i.e., a real–world system comprised of different kinds of objects and relations between them. A *complex network* depicts normally only one class of objects and one kind of relation between the instances of this object class. Examples for complex networks are the weblink graph, that depicts web pages and links connecting them, social networks, e.g., the hierarchical structures between employers of a company, or transport networks, i.e., certain places that are connected by means of transportation like streets or tracks. Thus, real–world or complex networks do not comprise a new kind of graph class that can structurally be differentiated from other graph classes. The term merely denotes those graphs that represent at least one real–world system in the above sense.

One of the structural properties of a graph is its *degree distribution*, i.e., the number of vertices with degree $deg(v) = k$ in dependence of the degree. It is well known that random graphs have a Poissonian degree distribution [9], with a mean degree of np and standard deviation of $\sqrt{np}$.

A *graph family* $\mathcal{G}_A(n, \Pi)$ is a set or graph defined by some algorithm A that gives a description to construct graphs for every given n and - if needed - an additional set of parameters Π. If A is a *deterministic algorithm* it constructs a single graph, if it is a *stochastic algorithm* it constructs all graphs with n vertices (and maybe additional parameters specified by Π) with a determined probability. The instance that is created from some defined graph family $\mathcal{G}_A(n, \Pi)$ is denoted by $G_A(n, \Pi)$. For graph families, we will often state expected properties with the words: *with high probability*, denoting that an instance $G_A(n, \Pi)$ constructed by A will show property X with a probability higher than $1 - 1/n$.

A *random graph* $G(n, p)$ is a graph with n vertices where every (undirected or directed) edge is element of E with probability p. A different but related algorithm for constructing random graphs is the $G(n, m)$ family of random graphs that picks m pairs of vertices and connects them with each other. Most of the time an implementation will try to avoid self-loops and multiple edges. Bollobás

states that for $\lim n \to \infty$ all expected properties of both families will be the same [9]. In the following, the term *random graph* will always denote an instance from the $G(n, p)$ model.

3 Data–driven, Mechanistic, and Game Theoretic Network Models

There are principally different ways of modeling real–world network data: one way tries to model one data set as closely as possible, resulting in a very accurate model of a given system. We call this the *data–driven approach* to network modeling. The other way searches for a structure that is common in many different systems, a so–called *universal structure*, and tries to explain the emergence of this structure with as few parameters as possible. These models follow the *mechanistic approach*. A third approach that is also often called a *network formation game* models complex systems in which networks emerge between independent agents. In most of these models the mechanism by which individual agents form bonds is given and the resulting network is in the focus of the analysis. In the following we will discuss these three types of network models and argue why and how mechanistic networks can be helpful in computer science.

Given an empirical data set, e.g., a social network that is explored by giving questionnaires to a group of people, it is necessary to model it as closely as possible while allowing some randomness. The random element allows for missing or false data which is often a problem in survey–based data and it also allows for deducing from the model the general behavior of other social networks with a similar overall structure. E.g., by exploring one email contact network in one company general statements might be possible about email contact networks in other companies with the same structure. These kind of graph models that try to describe a given empirical data set as closely and with as little parameters as possible can be called *data–driven graph models*. A very popular approach of this kind is the *exponential random graph model* [42, 43] and *block modeling* [39].

A very different perspective on modeling originates in the physics community: instead of modeling the details of a given graph they try to find an essential, mechanistic model that explains how a given structure could emerge in many different systems. These models are rather simplistic and will thus not provide very realistic models for any specific data set. But since they are so simplistic they can often be very easily adjusted to a specific system by adding back the peculiarities of that system.

Note that game–theoretic models of network formation are also very popular, but their perspective is in a way opposite to that of data–driven and mechanistic network models [7]: they define some kind of mechanism that determines how so–called agents decide which edges to build. The main question of these models is then which kind of network structures are stable in the sense that none of the agents would prefer to alter its own edge set. Thus, whereas data–driven and mechanistic network models start from a given structure that is modeled,

game–theoretic approaches start with a presumed network formation mechanism and analyze the resulting network structure.

In this chapter we will concentrate on *mechanistic graph models* that explain universal network structures with the least number of parameters necessary to produce them. Since these models are so simple but the resulting networks still show some very helpful properties, they can easily be adapted to be used in different computer–aided networks, e.g., P2P– or sensor–networks. We will thus concentrate on these mechanistic models in the rest of the chapter. We start by discussing the most common universal network structures and sketching some of the simple, mechanistic models that describe a reasonable mechanism for their emergence.

4 Real–World Network Structures and their Mechanistic Network Models

Among the structures found so far, the following four seem to be quite universal in many kinds of complex networks:

1. Almost all complex networks are *small–worlds*, i.e., locally they are densely connected while the whole graph shows a small diameter [48] (s. Subsect. 4.1);
2. Most of all complex networks are *scale–free*, i.e., most vertices have a low degree but some have a very high degree [6] (s. Subsect. 4.2);
3. Many of them are *clustered*, i.e., it is possible to partition the graph into groups of dense subgraphs whose interconnections are only sparse [16, 23, 37, 40];
4. At least some of them seem to be *fractal* [44].

Other structures, especially small subgraphs in directed networks, so–called *network motifs*, have been identified in only a few networks, especially biological networks [35, 36]. We will concentrate on the first two structures since it has already been shown that these properties influence processes on networks that are relevant in computer science as we will sketch in the following.

4.1 Small–Worlds

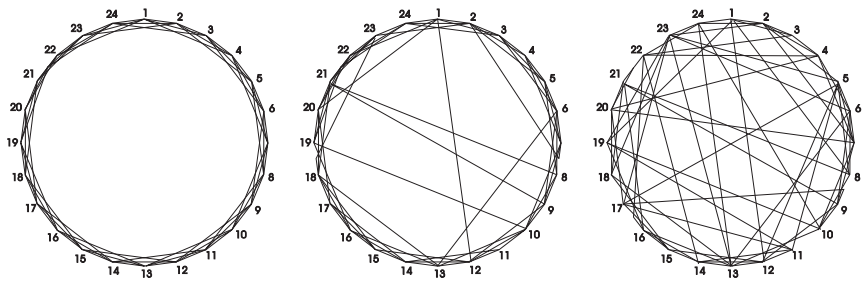
In 1998, Watts and Strogatz reported that in many real–world networks the vertices are locally highly connected, i.e., *clustered*, while they also show a small average distance to each other [48]. To measure the clustering they introduced the so–called *clustering coefficient* $cc(v)$ of a vertex v to be:

$$cc(v) = \frac{2e(v)}{deg(v)(deg(v)-1)}, \tag{2}$$

where $e(v)$ denotes the number of edges between all neighbors of v[4]. Since the denominator gives the possible number of those edges, the clustering coefficient

[4] The clustering coefficient of vertices with degree 1 is set to 0 and it is assumed that the graph does not contain isolated vertices.

of a single vertex denotes the probability that two of its neighbors are also connected by an edge. The *clustering coefficient* $CC(G)$ of a graph is the average over all the clustering coefficients of its vertices. A high average clustering coefficient in a graph seems to indicate that the vertices are connected locally and that thus the average distance of the according graph will be large. This is, e.g., the case in a simple grid, where vertices are only connected to their next four neighbors and thus the diameter scales with $\sqrt{n}$. Astonishingly, the diameter of the real–world networks analyzed in the paper of Watts and Strogatz was more similar to that of a corresponding random graph from the $G(n, p)$ model, despite their high average clustering coefficients. A random graph is said to be *corresponding* to a real–world network if it has the same number of vertices and expectedly the same number of edges. Such a graph can be achieved by setting p to $2m/(n(n-1))$. Of course, the expected clustering coefficient of vertices in such a graph will be p since the probability that any two neighbors of vertex v are connected is p. It now turned out that the real–world networks the authors analyzed had a clustering coefficient that was up to $1,000$ times higher than that of a corresponding random graph.



(a) $G_c(24, 3)$, $p = 0$ (b) $p = 0.1$ (c) $p = 1$

Fig. 1. a) A circulant graph of 24 vertices where each vertex is connected to its three nearest neighbors. **b)** Each edge has been rewired with probability $p = 0.1$. **c)** Each edge has been rewired with $p = 1.0$.

The first mechanistic model to reproduce this behavior, i.e., a high average clustering coefficient combined with small average distance, was given by Watts and Strogatz (s. Fig. 1): They start with a set of n vertices in a circular order where each vertex is connected with an undirected edge to its k clockwise next neighbors in the order. Such a graph is also called a *circulant graph*. Each edge is subsequently *rewired* with probability $0 \leq p \leq 1$. To *rewire* an edge $e = (v, w)$ a new target vertex w' is chosen uniformly at random, and (v, w) is replaced by (v, w'). It is clear that with $p = 0$, the clustering coefficient is given by:

$$CC(G) = \frac{3(k-1)}{2(2k-1)}, \qquad [38, p.289] \tag{3}$$

which approaches 3/4 in the limit of large k. The average distance between pairs of vertices is $n/4k$ which scales linearly with the number of vertices. If now the average clustering coefficient and the average distance is analyzed with respect to p (s. Fig. 2), it is easy to see that the average distance drops much faster than the average clustering coefficient. Thus, for small p, the average clustering coefficient is still quite high while the average distance has already dropped to a value comparable to that of a corresponding random graph. The regime in which that happens defines the set of networks showing the small–world effect or, for short: the set of small-world networks. Note that this very blurry definition has never been stated more rigorously. As Newman, Watts, and Barabási put it, nowadays the term "small–world effect" has come to mean that the average distance in the network increases at most (poly–)logarithmically with n while the average clustering coefficient is much higher than $p = 2m/(n(n-1))$.

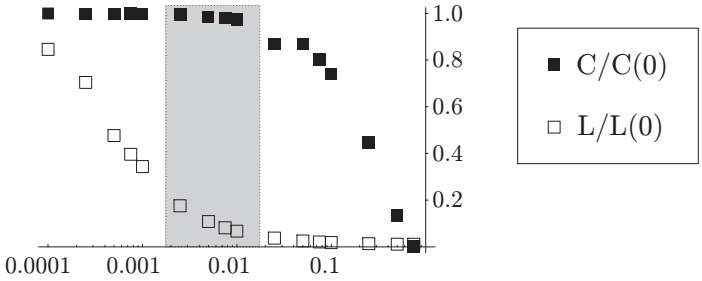


Fig. 2. Shown is the average clustering coefficient (denoted by C here) and the average distance (L) in the Watts–Strogatz–model, in dependence of the rewiring probability p and normalized by the resulting value for $p = 0$ ($C(0)$, $L(0)$). The average clustering coefficient is quite stable as long as not too many edges are rewired, while the average distance drops very fast, even at very low values of p.

Next to this rewiring model, other authors suggested to model small–world networks by composing something like a local, grid–like graph with a very sparse random graph, e.g., Kleinberg [26, 25] and Andersen et al. in their hybrid–graph model [4, 14]. The simplest model is to take a d–dimensional grid graph on n vertices and additionally connect each pair of vertices with probability p. This hybrid graph model is denoted by $G_d(n, p)$. Of course, if p is around $(\log n)^{1+\epsilon}/n$ for some constant $\epsilon > 0$ the edges constituting the random graph part alone will induce an average distance of $O(\log n)$ [10]. As we could show, also much smaller values of p suffice to reduce the diameter (and thus the average distance) to a poly-logarithmical term:

Lemma 1. *[29, 49] For $p = \frac{1}{cn}$, $c \in \mathbb{R}^+$ and $\epsilon > 0$ the diameter of $G_d(n,p)$ is asymptotically bound with high probability by at most*

$$d \cdot \left(\left\lceil \sqrt[d]{c \cdot (\log n)^{1+\epsilon}} \right\rceil - 1 \right) \cdot \left(\frac{\log n}{(1+\epsilon)\log\log n - \log 2} + 1 \right). \tag{4}$$

Very broadly speaking: the diameter of a combined graph is linearly dependent on the dimension d of the underlying grid and on (very broadly) $O(\log n^{1+(1+\epsilon)/d})$.

This result can be generalized as follows:

Lemma 2. *[29, 49] For any function $(\log n)^{-(1+\epsilon)} \leq f(n) \leq n^{1-\delta}, \epsilon, \delta > 0$ and $p = \frac{1}{f(n) \cdot n}$ the diameter of $G_d(n,p)$ approaches asymptotically with high probability*

$$d \cdot \left(\left\lceil \sqrt[d]{f(n) \cdot (\log n)^{1+\epsilon}} \right\rceil - 1 \right) \cdot \left(\frac{\log(n/f(n))}{\log(\log n)^{1+\epsilon}} \right). \tag{5}$$

Roughly speaking, if p is set to $1/(n \log n)$, i.e., only every $\log n$-th vertex is incident with a random edge, the average distance in the $G_d(n,p)$ model for large n and $d = 2$ is given by $O(\sqrt{\log^{2+\epsilon} n} \log n)$, i.e., $O(\log^{2+\epsilon/2} n)$. This result is important for network design: let us assume that the cost for connecting two communication devices scales with their distance. If only connections up to a certain distance are built, the diameter of the resulting network will scale approximately linearly with the largest distance. Our result now shows that only a small amount of random–like edges has to be built in order to shrink the network's diameter to a quadratic logarithmic term.

It could be shown that small–worlds are ubiquitious. Especially important for computer scientists, all technical communication networks, like the Internet and various overlay–networks, have the property that they show a small average distance together with a high clusteredness. Thus, if a new kind of Internet protocol or a new P2P–system is simulated, it is very important to simulate them on a network model that comprises these two properties.

As stated above, the small–world effect assumes that the average distance is bound by $O(\log^k n)$ for some constant k. The models given above will essentially show a growing average distance for increasing numbers of n. In 2005, Leskovec et al. analyzed the average distance in growing real–world networks [30, 31]. They found that actually the diameter of growing graphs *shrinks* in some cases instead of growing (poly–)logarithmically with increasing n. They also present some new network models that describe this behavior. We refer the interested reader to their paper [30].

Another, even more important structural property of networks was found in 1999, the so–called *scale–freeness* of real–world networks.

4.2 Scale–Free Networks

Given a graph G let $P(k)$ denote the probability to choose a vertex with degree k uniformly at random from all vertices of the graph. A complex network is said to be scale–free when $P(k)$ scales with $k^{-\gamma}$ where γ is a positive constant [6].

This distribution is also said to follow a *power law*[5]. For real–world networks, γ is often between 2 and 3, see [38, Table 3.7] for an overview of n, m, and γ of around 30 different network types. The easiest way to detect a scale–free distribution is to compute $P(k)$ and to display it in dependence of k in a double logarithmic diagram. Since $\log P(k) \simeq -\gamma \log k$ the resulting plot shows a straight line. Note however that if the available data does not span a large order of magnitudes it is actually hard to differentiate between power–laws and other possible distributions. Clauset et al. describe how to make more accurate estimates of the parameters governing a power–law distribution [17]. One such scale–free degree distribution in a real–world complex network was observed by the Faloutsos brothers in 1999 when they analyzed a sample of the weblink graph [20].

As is obvious by the construction of a random graph, its degree distribution follows a normal distribution (for large n). In comparison with this, a scale–free graph with the same number of vertices and edges as a corresponding random graph has much more low–degree vertices and also some vertices with a much higher degree than to be expected in a random graph. These high–degree vertices are also called *hubs*.

The first model that explained how such a scale–free degree distribution can emerge is the *preferential attachment* or *the–rich–get–richer* model by Barabási and Albert [6]: it is a dynamic network model where in each step i one new vertex v_i is added together with k incident edges. It starts with a small random graph of at least k vertices. Subsequently, each new vertex v_i chooses k vertices from the already existing ones, each with a probability that is proportional to its degree at this time point and creates an edge to it. More precisely, the probability that the newly introduced vertex v_i chooses vertex w is proportional to $deg(w)$ at that time point. Thus, if a vertex already has a large degree, it has a higher chance to get a new edge in each time step, a process which is called *preferential attachment*. This procedure produces a scale–free network in the limit of large n [1]. Note that the original model is not defined in every detail, especially the starting graph and the exact choice mechanism are not fully defined. As Bollobás and Riordan point out, different choices can lead to different results [10]. Their LCD model is defined rigorously and can thus be analyzed more easily [11]. For an overview on other scale–free producing network models see also [7].

Many real–world networks are reported to have scale–free degree distributions, the Internet [20], the weblink graph [6], or the web of human sexual contacts [32, 33], to name just a few. Especially the findings on the sexual contact network have important consequences: first, it can explain why sexual diseases are so easily spread and second, it can also help to prevent the spreading, by finding especially active hubs and treat them medically. This follows from the fact that scale–free networks are most easily disconnected if only a small fraction of the high–degree vertices are removed as we will discuss in Section 4.3.

[5] Note that power laws have different names like 'Lotka's law' or 'Zipf's law' and were investigated much earlier in other disciplines. E.g., 1926 Lotka observed that citations in academic literature might follow a power law [34].

The scale–free nature of email contact networks can also explain why some (computer) viruses stay nearly forever in the network: Pastor-Satorras and Vespignani discuss a model of virus spreading over a scale–free network and show that in this network there is no *epidemic threshold*, i.e., no minimal infection density to enable the infection of nearly the whole network [41]. Thus, in such a network, every virus can potentially infect the whole network.

4.3 Robustness of Random and Scale–Free Networks

One very interesting finding of Albert et al. is that different network structures show very different robustness against random failures and directed attacks against their structure [2]. They defined the *robustness* of a network as the average distance after a given percentage of the vertices were removed from the network. The removal is modeled in two ways: to model a random failure of, e.g., a server in the internet, any vertex is chosen uniformly at random to be removed from the network; to model a directed attack of some malicious adversary that knows the network structure, the vertex with highest degree is removed from the network. Albert et al. could show that in a random failure scenario the robustness of a scale–free network is much higher than that of a corresponding random graph. Furthermore, after removing more and more vertices, the random graph finally disconnects into many small connected components while most vertices in the scale–free network are still forming a big connected component. But for an attack scenario, the reverse is true: while the random graph stays connected and shows a rather low average distance, the scale–free network will decompose after just a few high–degree vertices are removed. This behavior is easily explained: in a random failure scenario, most of the removed networks in a scale–free network will have a very small degree since most vertices in a scale–free network have a small degree. In a random graph, almost all vertices have the same degree and the same importance for the connectedness of the graph. This property saves the network in the case of directed attacks. But the scale–free network will lose a high percentage of its edges very quickly if its high–degree vertices are removed which makes it very vulnerable to this kind of attack. This result, although very easily explained when it was discovered, is quite devastating since most of our communication and even some of our transportation networks, especially flight networks, are scale–free. Thus, Albert et al. showed how easily these networks might become disconnected. We will show in the following section that this problem can be alleviated by allowing the network to react to the attacks.

5 Network Design for Systems with Independent Agents

In many complex systems there is no central authority, they are decentrally organized. In these networks, it is, e.g., very hard for a single participant to understand whether a missing neighbor is missing due to a random failure or a directed attack since this requires a global overview. Nonetheless, because of the different robustness of random and scale–free network structures (s. Subsec. 4.3)

it would be very convenient if a network could change its structure according to the situation it is in, i.e., to a random network in the case of attacks and to a scale–free network in the case of random failures. In the following section we will show that this can be achieved in a decentrally organized network. In Sec. 5.2 we will then show how carefully a network generating protocol has to be implemented in a decentrally organized system because a subtle change can make the difference between an efficient and an inefficient network evolution.

5.1 Adaptive Network Structures

In [50] we considered the question of whether a decentrally organized network can adapt its network structure to its environmental situation, i.e., a random failure scenario or an attack scenario, while the single participants are oblivious of this situation. The first thing to observe is that it is not necessary to have a really random network structure in the case of attacks, it suffices to have a network in which almost every vertex has the same degree. It is also not necessary to make a real scale–free network in which the degree distribution is described by $P(k) \simeq k^{-\gamma}$. It suffices if the degree distribution is sufficiently right–skewed, i.e., if most of the vertices have a low degree and some a very high degree. The second observation to be made is that in the case of attacks, the wanted change towards a more uniform degree distribution is essentially achieved by the nature of the attack itself: since it removes high–degree vertices it smoothes the degree distribution. A third observation is that in the model of Albert et al. the vertices are not allowed to react to the situation. Thus, in the scale–free network the attack on only a few high–degree vertices already removes a high percentages of the edges. We will now present a network re–generating protocol that is decentral and oblivious of the situation in which the network is in and achieves to adapt the network's structure to the one best suited for the situation. Consider the following model: In each time step remove one vertex x at random (random failure scenario) or remove the vertex with the highest degree (attack scenario). A vertex v notices if its neighbor x is missing. Vertex v will now build one edge to any of its neighbors w in distance 2 with probability 0.5. Let this set of neighbors in distance 2 be denoted by $N_2(v)$. The probability with which a vertex v chooses w is computed by the following generic formula:

$$p_i(v,w) = \frac{deg(w)^i}{\sum_{w' \in N_2(v)} deg(w')^i}. \tag{6}$$

Thus, if $i = 0$ all second–hand neighbors have the same probability to be chosen, if $i = 1$ the process resembles a local preferential attachment. Algorithm 1 describes this re–generating behavior in pseudo code. In the following $A0$ will denote this algorithm where $p_i(v,w)$ is computed with $i = 0$ and $A1$ denotes the algorithm where i is set to 1.

Note that to be able to compare the robustness of the resulting networks it is necessary to keep the number of vertices and edges constant. Thus, the removed

Algorithm 1 Algorithm for rewiring a deleted edge to one of the second neighbors.

```
procedure NODE.REWIRE(NODE v)                                          ▷
    if (any neighbor of node v is deleted) then
        if (random.nextDouble() < 0.5) then
            target ← choose second neighbor w with probability P_i(v, w));
            create edge between node v and target;
        end if
    end if
end procedure
```

vertex x is allowed to re–enter the network, building edges at random to other vertices. To keep the number of edges (at least approximately) constant x will build half as many edges as it had before the removal[6]. Thus, since every of its former neighbors builds a new edge with probability 0.5 and itself builds another $deg(x)/2$ edges, the number of edges stays approximately constant.

Consider now a random graph that suffers from random failures. In this scenario, we would like the graph to establish a right-skewed degree distribution. Fig. 3 shows that algorithm $A1$ is faster in building a right–skewed degree distribution than $A0$. This implies that also a local preferential attachment is enough to create something like a scale–free degree distribution.

Remember that a real scale–free network structure stabilizes the network against random failures, i.e., it keeps a low average distance even if a substantial part of the vertices are removed at random. It also makes the network more fragile against directed attacks. To measure this effect we use Albert et al.'s definition of robustness: the *robustness* of a graph as introduced by Albert et al. [2] is measured by the *average distance between all vertices* after a given percentage of nodes is removed from the graph (without any rewiring). In the following we will set this value to 5%. If robustness against attacks is measured, the removed vertices are the 5% vertices with highest degree, in the case of random failures the set is chosen uniformly at random. The robustness measures are denoted by $\mathcal{R}_A(G)$ for attack robustness and by $\mathcal{R}_{RF}(G)$ for random failure robustness. Note that this measure as introduced by Barabási and Albert is a bit unintuitive since a *higher* value denotes a *less* robust network and a *lower value* denotes a *more* robust network.

To analyze whether $A1$ creates a robust network against random failures and a fragile network with respect to attacks, we started with a random graph with $1,000$ vertices and $5,000$ edges. This network then experiences $20,000$ random failures. After $1,000$ failures each, the resulting graph is taken and $R_A(G)$ and $R_{RF}(G)$ are measured. After that, the next $1,000$ random failures are simulated together with the re–generating algorithms $A0$ and $A1$, respectively.

[6] We tried many different variations of re–inserting the removed vertex. The general outcome did not seem to be influenced by the details of this procedure. It should be noted that the method sketched here neither introduces a skewed nor a very narrow degree distribution on its own.

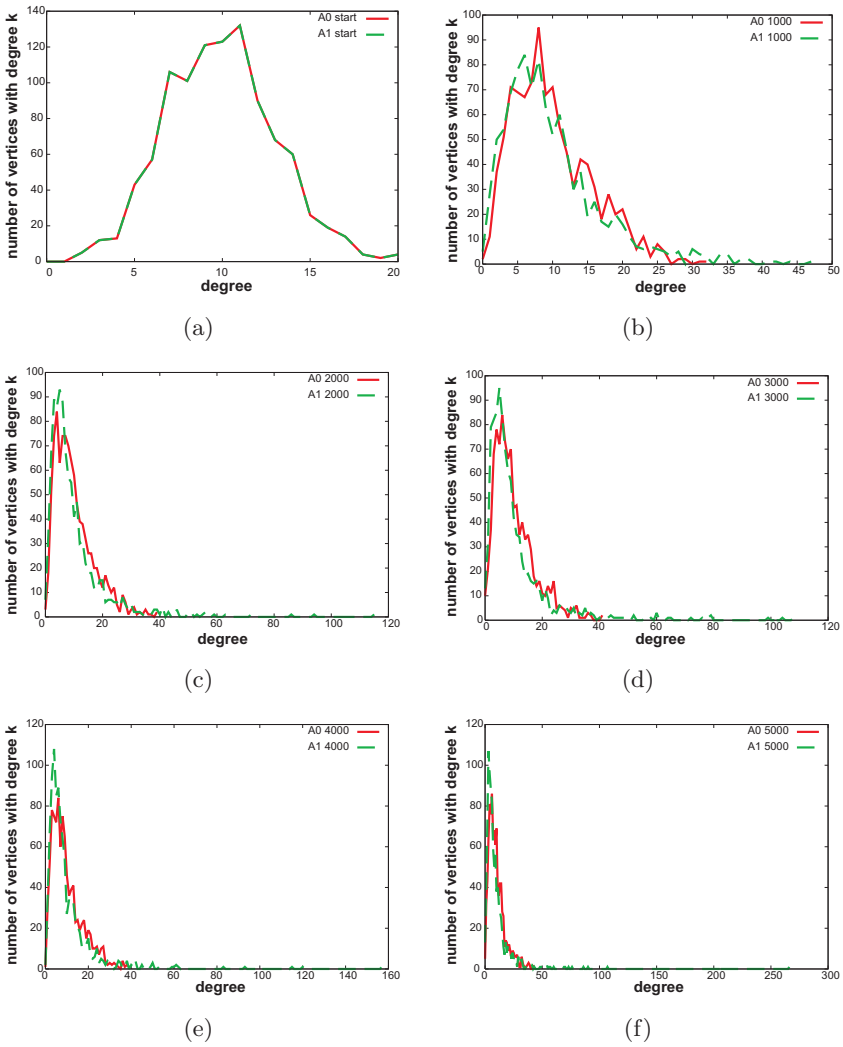


Fig. 3. Exemplary evolution of the degree distribution of one random graph after $5,000$ random failures, plotted after every 1000 deletions. (a) Since both runs start with the same random graph with $n = 1000$ and $m = 5,000$, the degree distribution is the same for both algorithms. (b)-(f) Each diagram compares the resulting degree distribution after applying algorithm $A0$ and $A1$. It is clear to see that $A1$ results in a degree distribution that is more right-skewed than the one created by $A0$. For example, in diagram f, the highest degree in the graph resulting from procedure $A1$ is around 250, that of $A0$ is around 50.

In a pure random graph with $1,000$ vertices and $5,000$ edges $R_A(G)$ is 3.4 and an $R_{RF}(G)$ is 3.3, i.e., as expected the increase in the average path length

is very similar and only slightly higher in the attack scenario than in the random failure scenario. In a pure scale–free network with the same number of vertices and edges, R_A is higher than that in a random graph, namely 3.5. As expected, the robustness against random failures in a pure scale–free graph is much better than that of the random graph with $R_{RF}(G)$ being 3.0. Thus, we expect that after some random failures the robustness measures after applying $A0$ and $A1$ should match that of a pure scale–free graph, i.e., approach 3.5 for $R_A(G)$ and 3.0 for $R_{RF}(G)$ (s. Fig. 4).

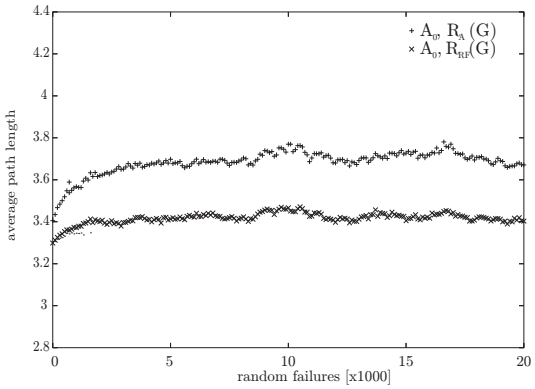
	Random Graph	Scale–Free Graph	$A0$ after $20,000$ steps	$A1$ after $20,000$ steps
$R_A(G)$	3.4	3.5	3.6	3.9
$R_{RF}(G)$	3.3	3.0	3.4	3.1

Table 1. Comparison of the robustness of pure random graphs and pure scale–free networks with the networks that result after applying $20,000$ random failures and either regenerating algorithm $A0$ or $A1$. It is clear to see that the network resulting with algorithm $A0$ performs worse than both the pure random and the pure scale–free graph, while the graph resulting from $A1$ comes near to the robustness of a pure scale–free graph in the case of attacks. On the other hand, it is even more fragile in the case of directed attacks. This is due to the localized structure of the network regenerating algorithm.

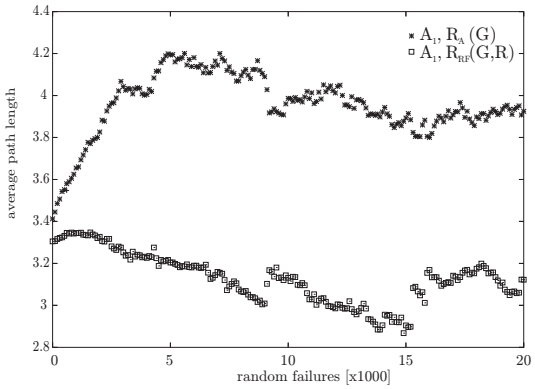
Algorithm $A0$ does not achieve this goal, even after $20,000$ random failures its robustness in both cases, random failures and attacks, is worse than that of a pure random graph or a pure scale–free graph (s. Table 1). Algorithm $A1$ produces a network that is nearly as robust against random failures as the pure scale–free graph which is astonishing since it only uses local information. On the other hand it produces networks that are even more fragile with respect to attacks than pure scale–free graphs. But, as we could show, this is not a huge problem since the nature of the attack will very quickly move the network's degree distribution back to a narrow and thus robust distribution.

In summary, this model shows that the fragility of scale–free networks can be alleviated if the network is allowed to react to missing vertices. Of course this is not always possible on a short time–scale, e.g., in street networks. But for example in flight networks, a redirection of airplanes is only a minor problem. Also in P2P–networks a new edge can be built fast and without high costs.

As we have shown in the previous paragraphs, many relevant technical networks show specific network structures that influence the behavior of processes running on top of them. To analyze the behavior of a new communication protocol or the runtime of a graph algorithm on real–world networks, it is very important to analyze the network's structure and to model it as closely as possible. But sometimes computer scientists face the opposite perspective, namely to design a network such that its structure supports the processes on top of it.



(a) Robustness of graphs resulting from A_0 in a random failure scenario



(b) Robustness of graphs resulting from A_1 in a random failure scenario

Fig. 4. Evolution of $\mathcal{R}_A(G)$ and $\mathcal{R}_{FT}(G)$ in a long random failure scenario with $20,000$ events and application of $A0$ and $A1$. Starting graph is a random graph with 1000 vertices and 5000 edges. (a) Algorithm $A0$ creates a graph that is less robust against attacks than a pure random graph: its average path length after removing 5% of the vertices is 3.6 compared to 3.4 in a pure random graph. It is also higher (less robust) than the value in a pure scale–free graph which has $\mathcal{R}_A(G) = 3.5$. The graph's robustness against random failures is worse than that of a pure random graph (3.4 vs. 3.3). (b) As expected, algorithm $A1$ is able to create a graph that is at least as robust against random failures as a comparable scale–free graph ($\mathcal{R}_A(G) \simeq 2.9 - 3$ compared to 3.0 of a pure scale–free graph). Accordingly, its robustness against attacks is even worse than a comparable scale–free graph ($\simeq 4$ vs. 3.5), i.e., the resulting graph's short paths are strongly depending on the high degree vertices. Note that jumps in the curves are caused by deletion of a high degree vertex by chance.

This is already a difficult task if the network is static and centrally organized, but nowadays it becomes more and more important to design network generating rules between independent and maybe even mobile agents, like in sensor networks, peer–to–peer networks [28], or robot swarms. In the following section we will show how difficult this design task can be, even in a toy example.

5.2 The Sensitivity of Network Generating Algorithms in Decentrally Organized Systems

Social systems are among the most complex systems to explore because their global structure depends on the individual decisions of the humans that constitute them. We can assume that humans will interact with each other when both profit from this interaction in a way, i.e., we will assume that the human agents are selfish. In internet–based communication networks, foremost P2P–networks, these decisions are mediated by the software that manages the communication over the internet. A P2P–network can be seen as an overlay network of the internet, i.e., every user has a *buddy list* of other participants with which she can communicate directly. If a query, e.g., for a file, cannot be answered by one of her neighbors, the query will be redirected to (all or a choice of) the neighbors of her neighbors and so on. The complex network of this system thus represents each participant p_i as vertex v_i, and v_i is connected to those vertices that represent the participants on the buddy list of p_i. As already sketched above, some network structures are more favourable than others. For example, it might be wanted that a P2P–network has a small diameter. This sounds like a network property that has to be managed centrally. On the other hand, we know that if every participant had only one random edge the diameter would already scale poly–logarithmically. But why should a random edge be valuable for a single participant? In the example of P2P–networks, especially file–sharing networks, it is much more valuable to be connected to those participants that have a similar interest than to any random participant which may never have any interesting file to share. Thus, the question is: what kind of control can be excerted to guide the decisions of each single user such that an overall favourable network structure arises? Especially in software mediated communication networks, the designer can indeed excert some control, namely by the information about the current network structure the software feeds back to the users. Consider the following toy example: The initial starting configuration is a tree. Each participant v is told its *eccentricity* $ecc(v)$, i.e., its maximal distance to any other participant in the network:

$$ecc(v) := \max_{w \in V} d(v, w). \tag{7}$$

Given this information, a vertex is *satisfied* if this eccentricity does not exceed some constant k. In each step, one participant v is chosen at random. If its eccentricity is larger than k, it tries to improve its position in the network: to do so, let $N_2(v)$ denote the vertices in distance 2 from v that are no leaves. Now, v chooses one of these vertices z uniformly at random. Let w denote the mediating neighbor of v and z (s. Fig. 5).

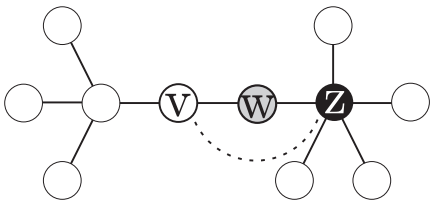


Fig. 5. One vertex v is chosen at random in every time step. If its eccentricity is greater than k it will try to connect to a non-leaf vertex in distance 2 (black vertex). Let z be the second neighbor chosen and w be the vertex connecting both. Then edge (v, w) will be replaced by edge (v, z) if the eccentricity of v does not increase due to this process.

The edge (v, w) is then temporarily removed and the edge (v, z) is temporarily built. If the eccentricity of v does not increase by this new edge, v will keep the edge and otherwise the original edge (v, w) is re–established. It can be shown that this procedure will eventually form a tree with diameter k, i.e., a tree in which all participants are satisfied [49, Lemma 6.1]. Of course, it is important how fast this globally satisfying structure will be achieved. Let now k be equal to 2, i.e., the globally satisfying network structure is a star. Unfortunately, the following theorem shows that this will take expectedly exponential time:

Theorem 1. *[27, 49, Theorem 6.1]*
Starting with a tree with diameter larger than 2*, the expected runtime to generate the star with the above sketched mechanism is bounded from below by* $\Omega(2^n)$.

This situation can be greatly improved by feeding back the *closeness* $close(v)$ of each participant instead of its eccentricity:

$$close(v) := \sum_{w \in V} d(v, w). \tag{8}$$

By assuming that each participant will only accept a new edge if its closeness is strictly decreased, it can first be shown that a globally satisfying network is finally achieved [27, 49, Lemma 6.2]. Furthermore, the expected time until this happens is polynomial:

Theorem 2. *[27, 49, Theorem 6.2]*
If the closeness is fed back to each participant, the expected runtime until an overall satisfying network structure is generated is bounded by $O(n^5)$.

By a more involved analysis, Jansen and Theile [24] improved the upper bound to $O(n^{5/2})$ [Th. 4] and could show that the runtime is bounded from below by $\Omega(n \log n)$ [Th. 3]. Of course, this toy model of a complex system generated by independent, selfish agents is not easily applied to realistic situations, in which

we have to deal with graphs instead of trees, in which agents change the network at the same time (asynchronous behaviour), and in which it is also unlikely to be able to compute the closeness or eccentricity. But it proves the following two points:

1. Even in a system comprised of independent agents, a designer can excert some control over the network generating process by feeding back a well–designed subset of structural information.
2. The choice of which information to feed back must be made very carefully since it can make the difference between an exponential and a polynomial runtime until a satisfying network structure is achieved.

6 Summary

The last years have shown that real–world networks have a distinct structure that is responsible for the behavior of various processes that take place on them. We have cites some papers that indicate that these structures also influence the behavior of technical networks, especially the internet or peer–to–peer networks. It could additionally be shown empirically that a small–world structure is likely to change the runtime of algorithms solving NP-hard problems on these graphs [45]. On the other hand, the complexity of, e.g., coloring a graph is unchanged on scale–free networks [21]. In any case, we think that further research should be directed to find and analyze those real–world structures that can be used to design more efficient algorithms or that make a problem even harder to solve in practice. As cited above, scale-free networks show a distinct behavior in computer simulations for, e.g., virus spreading, and thus simulations of similar processes should be based on an appropriately chosen network model that captures the essential structures of the according real–world network. Last but not least, computer scientists are more and more asked to design communication networks between humans and large swarms of mobile and independent devices. We have shown that it is indeed possible to control the network generating process even in a decentrally system of independent agents to achieve various, globally satisfying network structures, but we have also shown that a careful network generating protocol design is needed to do so. In summary, complex network science is important for computer scientists, as well in algorithm engineering as in the design of technical networks.

7 Further Reading

As sketched in Sec. 1, the field of complex network science can be divided into three areas: analysis, models, and processes on complex networks. At the moment there is no textbook that comprises all of these fields. We will thus refer to some review articles or books in each of the fields.

Network analysis was done long before the 1990s, especially in the social sciences. The textbook by Wasserman and Faust [46] is a classic book in that

realm. The book edited by Brandes and Erlebach covers most of these older and some of the newer results and moreover makes the effort to present them in a well–defined, formal framework [13]. The classic book on the analysis of random graph models is of course the one by Bollobás [9]. A new one that takes other, more recent random graph models into account is that by Chung and Lu [15].

Also the later results that are predominantly published by physicists have not yet found their way in one, comprehensive textbook. The early book by Dorogovtsev and Mendes covers almost only scale–free networks [18]. It is helpful since it explains some of the physics models used in this approach quite nicely. Watts has published his Ph.D. thesis which covers small–world network models [47]. Albert and Barabási have published a long review article (based on Albert's thesis) which is very readable and covers mostly scale–free network models and their behavior [1]. An interesting overview of applied complex network science is given in a handbook edited by Bornholdt and Schuster [12]. A very good collection of original papers, partitioned into five chapters alongside with a comprehensive introduction to each, was edited by Barabási, Watts, and Newman [38]. Alon has published a book that covers his findings on patterns in biological networks [3].

To our knowledge there is no comprehensive article or book that covers the behavior of processes on networks in dependency of their structure. Thus, we refer the reader to the last chapter of the article collection by Barabási, Watts, and Newman where at least some of this work can be found [38].

References

1. Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Review of Modern Physics*, 74:47–97, 2002.
2. Réka Albert, Hawoong Jeong, and Albert-László Barabási. Error and attack tolerance of complex networks. *Nature*, 406:378–382, 2000.
3. Uri Alon. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman & Hall/CRC, 2006.
4. Reid Andersen, Fan Chung, and Lincoln Lu. Analyzing the small world phenomenon using a hybrid model with local network flow. In *Proceedings of the WAW 2004*, LNCS 3243, pages 19–30, 2004.
5. Albert-László Barabási. *Linked - The New Science of Network*. Perseus, Cambridge MA, 2002.
6. Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.
7. Nadine Baumann and Sebastian Stiller. *Network Analysis: Methodological Foundations*, chapter Network Models, pages 178–215. Springer-Verlag, 2005.
8. Tim Berners-Lee, Wendy Hall, James A. Hendler, Kieron O'Hara, Nigel Shadbolt, and Daniel J. Weitzner. A framework for web science. *Foundations and Trends in Web Science*, 1(1):1–130, 2006.
9. Béla Bollobás. *Random Graphs*. Cambridge Studies in Advanced Mathematics 73. Cambridge University Press, London, 2nd edition, 2001.
10. Béla Bollobás and Oliver M. Riordan. *Handbook of Graphs and Networks*, chapter Mathematical results on scale-free random graphs, pages 1–34. Springer Verlag, Heidelberg, 2003.

11. Béla Bollobás and Oliver M. Riordan. The diameter of a scale–free random graph. *Combinatorica*, 24(1):5–34, 2004.
12. Stefan Bornholdt and Heinz Georg Schuster, editors. *Handbook of Graphs and Networks*. WILEY-VCH, Weinheim, 2003.
13. Ulrik Brandes and Thomas Erlebach, editors. *Network Analysis - Methodological Foundations*. Springer Verlag, 2005.
14. Fan Chung and Linyuan Lu. The small world phenomenon in hybrid power law graphs. In *Complex Networks (E. Ben-Naim, H. Frauenfelder, Z. Toroczkai (eds.))*, pages 91–106, 2004.
15. Fan Chung and Linyuan Lu. *Complex Graphs and Networks*. American Mathematical Society, 2006.
16. Aaron Clauset, Mark E.J. Newman, and Christopher Moore. Finding community structure in very large networks. *Physical Review E*, 70:066111, 2004.
17. Aaron Clauset, Cosma Rohilla Shalizi, and Mark E.J. Newman. Power–law distributions in empirical data. ArXiv, June 2007.
18. Sergei N. Dorogovtsev and Jose F.F. Mendes. *Evolution of Networks*. Oxford University Press, 2003.
19. Paul Erdős and Alfréd Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.
20. Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. *Computer Communications Review*, 29:251–262, 1999.
21. Alessandro Ferrante, Gopal Pandurangan, and Kihong Park. On the hardness of optimization in power-law graphs. *Theor. Comput. Sci.*, 393(1-3):220–230, 2008.
22. E. N. Gilbert. Random graphs. *Anual Math. Statist.*, 30:1141–1144, 1959.
23. Michelle Girvan and Mark E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99:7821–7826, 2002.
24. Thomas Jansen and Madeleine Theile. Stability in the self-organized evolution of networks. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, pages 931–938, 2007.
25. Jon Kleinberg. Navigation in a small world. *Nature*, 406:845, 2000.
26. Jon Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, pages 163–170, 2000.
27. Katharina A. Lehmann and Michael Kaufmann. Evolutionary algorithms for the self-organized evolution of networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'05)*, pages 563–570, 2005.
28. Katharina A. Lehmann and Michael Kaufmann. *Peer-to-Peer Systems and Applications*, chapter Random Graphs, Small Worlds, and Scale-Free Networks, pages 57–76. Springer Verlag, 2005.
29. Katharina A. Lehmann, Hendrik D. Post, and Michael Kaufmann. Hybrid graphs as a framework for the small-world effect. *Physical Review E*, 73:056108, 2006.
30. Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: Densification laws, shrinking diameters, and possible explanations. In *Proceedings of the 11th ACM SIGKDD*, 2005.
31. Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):No. 2, 2007.
32. Fredrik Liljeros. Sexual networks in contemporary western societies. *Physica A*, 338:238–245, 2004.

33. Fredrik Liljeros, Christofer R. Edling, Luís A. Nunes Amaral, H. Eugene Stanley, and Yvonne Åberg. The web of human sexual contacts. *Nature*, 411:907–908, 2001.
34. Alfred Lotka. The frequency distribution of scientific productivity. *Journal of the Washington Academy of Sciences*, 16:317–323, 1926.
35. Ron Milo, Shalev Itzkovitz, Nadav Kashtan, Reuven Levitt, Shai Shen-Orr, Inbal Ayzenshtat, Michal Sheffer, and Uri Alon. Superfamilies of evolved and designed networks. *Science*, 303:1538–1542, 2004.
36. Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298:824–827, 2002.
37. Mark E.J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences, USA*, 98(2):404–409, 2001.
38. Mark E.J. Newman, Albert-László Barabási, and Duncan J. Watts, editors. *The Structure and Dynamics of Networks*. Princeton University Press, Princeton and Oxford, 2006.
39. Marc Nunkesser and Daniel Sawitzki. *Network Analysis: Methodological Foundations*, chapter Blockmodels, pages 178–215. Springer-Verlag, 2005.
40. Gergely Palla, Imre Derényi, Illes Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, 2005.
41. Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic spreading in scale-free networks. *Physical Review Letters*, 86(4):3200–3203, 2001.
42. Garry Robins, Pip Pattison, Yuval Kalish, and Dean Lusher. An introduction to exponential random graph (p^*) models for social networks. *Social Networks*, 29:173–191, 2007.
43. Gary Robins, Tom Snijder, Peng Wang, Mark Handcock, and Philippa Pattison. Recent developments in exponential random graph (p^*) models for social networks. *Social Networks*, 29:192–215, 2007.
44. Chaoming Song, Shlomo Havlin, and Hernán A. Makse. Origins of fractality in the growth of complex networks. *Nature*, 2:275–281, 2006.
45. Toby Walsh. Search in a small world. In *Proceedings of the IJCAI-99*, 1999.
46. Stanley Wasserman and Katherine Faust. *Social Network Analysis - Methods and Applications*. Cambridge University Press, Cambridge, revised, reprinted edition, 1999.
47. Duncan J. Watts. *Small Worlds- The Dynamics of Networks between Order and Randomness*. Princeton Studies in Complexity. Princeton University Press, 1999.
48. Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.
49. Katharina A. Zweig. *On Local Behavior and Global Structures in the Evolution of Complex Networks*. PhD thesis, University of Tübingen, Wilhelm-Schickard-Institut für Informatik, 2007.
50. Katharina Anna Zweig and Karin Zimmermann. Wanderer between the worlds – self-organized network stability in attack and random failure scenarios. In *Proceedings of the 2nd IEEE SASO*, 2008.