Wanderer between the worlds - Self-Organized Network Stability in Attack and Random Failure Scenarios

Katharina A. Zweig
Eötvös Loránd University,
Department of Biological Physics,
1117 Budapest, Pazmany P. stny 1A, Hungary
nina@angel.elte.hu

Karin Zimmermann
University of Tübingen,
Wilhelm-Schickard-Institute,
Sand 14, 72076 Tübingen, Germany
zimmerm@informatik.uni-tuebingen.de

Abstract

Many real–world networks show a scale–free degree distribution, a structure that is known to be very stable in case of random failures. Unfortunately, the very same structure makes the network very sensitive against targeted attacks on their high-degree vertices. Under attack it is instead preferrable to have a Poisson- or normal degree distribution. The paper adresses the question of whether it is possible to design a network protocol that enables the vertices of a decentralized network to switch its topology according to whether it is attacked or just suffers of random failures. We further require that this protocol is oblivious of the kind of removal scenario, i.e., that is does not contain any kind of attack detection. In this article we show how to design such a protocol that is oblivious, needs only local information, and that is capable of switching between the two structures reasonably fast. The protocol is easy to implement and keeps the average degree in the graph constant. After an analytical and empirical evaluation of the result the paper is concluded by possible applications to internet-based communication networks.

1. Introduction

The modern world depends on a number of technical networks, and their understanding is more and more crucial to stabilize our economy. Starting with some empirical analysis on structural properties of real-world networks [27, 3, 15] it was soon clear that random graph models [4] are too simple to model the structural richness of the WWW [3], the Internet [6], social networks of various kinds [7, 16, 19], or peer-to-peer networks [13, 9]. This observation lead to a multiplicity of new network models, the classics being the so-called *small-world model* [27] and the *preferential attachment model* [3]. After these basic structural analysis methods and new network models were introduced, focus turned to analyze the behavior of processes on different kinds of networks, e.g., the cascading behavior of failures in power grids [18, 25], spreading of diseases [20], or navigation and routing [11, 12, 8]. One important finding concerned so-called *scale-free* networks, i.e., those where the probability $P(k)$ to find a vertex with k neighbors is proportional to $k^{-\gamma}$. Such a network is dominated by vertices with a small degree, but also contains some high degree vertices, so-called *hubs*. It could be shown that a scale-free network is very stable against random failures and at the same time very vulnerable to directed attacks on the high degree vertices [1]. It turned out that a random network where every vertex is connected with every other vertex with probability p is less stable than a scale-free network in the case of random failures, but is on the other hand more stable in the case of attacks. The differences are mainly based on the degree distribution: In a random graph every vertex has expectedly the same degree, and every vertex will contribute the same *stability* to the network. In the scale-free architecture, a high degree vertex that is attacked will severe the network's connectivity heavily, but on the other hand it is unlikely that a random failure will hit exactly this vertex.

Many of our modern communication networks, e.g., the Internet, the WWW, peer-to-peer networks, sensor networks, and other multi-hop communication networks, are prone to random failures and would thus benefit from a scale-free or at least right-skewed degree distribution. Indeed, some of the protocols of peer-to-peer networks naturally result in a scale-free network structure [22, 24]. On the other hand, these networks might suffer from attacks from time to time and in these situation it would be helpful to switch to a net-

work structure in which the degree distribution is normally distributed or is at least very narrow. Since most real-world networks are not centrally organized it is not possible to detect an attack situation by bird's eye view and change the network's structure in a concerted manner. One possibility is that every single vertex tries to detect an attack locally and subsequently changes its local neighborhood by connecting to random vertices in the network, as proposed by [10]. This approach is difficult: if every vertex can only see its local neighborhood it will detect an attack only if a large proportion of the network is already attacked. Here, we propose a second possibility, namely a simple reaction scheme that is able to drive the network's structure into the best possible structure independent whether it is in an attack or random failure scenario.

The paper is organized as follows: We give the necessary definitions in section 2, introduce the model in section 3, and its results in 4. The paper is finished by a discussion of related work and a summary in section 5.

2. Definitions

A *graph* is a pair (V, E), with V the set of vertices $\{v_1, v_2, \cdot, v_n\}$, and $E \subseteq V \times V$ the set of edges $\{e_1, e_2, \cdots, e_m\}$. Here, we will only regard *unweighted, undirected, single-edge, selfloop-free* graphs, i.e., every edge has the same length, an edge can be traveled in both directions, between all pairs of vertices there is at most one edge, and no vertex is connected to itself by an edge.

A *path* $P(v, w)$ between vertex v and w is a subset of consecutive edges $P(v, w) = \{e_1, e_2, \cdots, e_k\} \subseteq E$ with $e_1 = (v, v_1)$, $e_k = (v_{k-1}, w)$, and for all other edges $e_i = (v_{i-1}, v_i)$. The *length* $|P(v, w)|$ of a path is given by the number of edges in it. The *distance* $d(v, w)$ between any two vertices v, w is given by the *minimal length* of any path between them, and is ∞ by definition if there is no path between them. The *diameter* $\Delta(G)$ of a graph is defined as the maximal distance between any two vertices in the graph. The *average path length* $\Delta_\emptyset(G)$ is defined as the average over the distances of all pairs of vertices in the graph.

A *subgraph* $G' = (V', E')$ contains a subset of vertices V' of V and all edges $e = (v, w)$ with $v, w \in V'$. A *component* is a maximal subgraph where the distance between any two vertices is finite. The *size of a component* is defined as the number of vertices in it. The *biggest connected component* is defined as the component with the highest number of vertices in it.

The *set of neighbors* $N(v)$ contains all vertices with which it is connected by an edge, and in general $N_i(v)$ contains all vertices that are in distance i to v. Similarly, $E(v)$ denotes the set of edges that contain v.

The *degree* $deg(v)$ of a vertex v is defined as the number of its *neighbors*, i.e., $|N(v)|$. The *degree distribution* $deg(G)$ of a graph describes the number of vertices with degree k for all $0 \leq k \leq n$.

The *robustness* of a graph as introduced by Albert et al. [1] is measured by the increase in the *average distance between all vertices* after a given percentage of nodes is removed from the graph. 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.

A *random graph* $G(n, p)$ contains n vertices where every possible edge exists with probability p [4].

3. The Model

As sketched in the introduction, we want to find an algorithm that defines how to react when vertices are deleted from the network, either because they were attacked or because they just failed. An attack is modeled by deleting one of the vertices with highest degree uniformly at random, a random failure is modeled by deleting any vertex uniformly at random. We require that the algorithm that reacts to the deletions has the following properties:

1. The algorithm is *oblivious* of whether a vertex is deleted as the result of an attack or a random failure.

2. If the network is attacked, the degree distribution of the network will be driven towards a sharp distribution, centered around a mean value;

3. If the network experiences only random failures, the degree distribution is driven towards a right-skewed distribution;

4. The algorithm is run by all vertices and only local information is used, i.e., only information about vertices in the network of at most distance 2 to the executing vertex;

5. The average degree in the network is held (expectedly) constant;

Algorithm 1 presents a general algorithm that fulfills the above given requirements.

Essentially, every vertex v which misses a neighbor has a 50% chance of replacing the edge by an edge

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

```
1: procedure NODE.REWIRE                                   ▷
2:     if (any neighbor is deleted) then
3:         if (random.nextDouble() < 0.5) then
4:             target ← choose second neighbor w with
   probability P_i(v, w));
5:             create edge between this and target;
6:         end if
7:     end if
8: end procedure
```

to one of its second neighbors (line 4): the parameter i determines how strongly the degree of the second neighbor w influences the probability $P_i(v, w)$ that v creates a new edge to w. $P_i(v, w)$ is defined as:

$$P_i(v, w) = \frac{deg(w)^i}{\sum_{w' \in N_2(v)} deg(w')^i}, \tag{1}$$

If i is set to 1, this general form results in the preferential attachment probability as described in [3], but restricted to the neighbors in distance 2. If i is set to 0, every neighbor in distance 2 has the same chance to be chosen. Note that the denominator ensures that $P_i(v, w)$ describes a probability distribution that sums to 1.

It is easy to see that the algorithm is oblivious of the nature of a deletion and that it uses only local information. In the following we will show that also all other requirements are fulfilled.

We will first argue why the average degree is held expectedly constant: after the deletion of a vertex, the average degree should (expectedly) be the same, i.e.:

$$\frac{2m}{n} = \frac{2(m-x)}{n-1}, \tag{2}$$

where x denotes the number of edges that leave with a deleted vertex. Solving for x gives:

$$x = \frac{m}{n} = \frac{\deg_\emptyset}{2}. \tag{3}$$

Thus, if every deletion of the vertex decreases the number of edges by half of its degree, the average degree is expectedly constant.

We will now show that the algorithm is able to shift the degree distribution into a shape that is appropriate in the given situation. We will restrict this discussion to the cases of $i \in \{0, 1\}$ and call the respective versions of the algorithm A_0 and A_1. The first step is to show why a narrow degree distribution emerges in the case of attacks: if initally the network starts with a right-skewed degree distribution and is then attacked, the desired narrowing of the distribution is mainly organized by the nature of the attack itself: since the attacker does not allow any vertex to have a much higher than average degree, only vertices with at most average degree stay in the network. Thus, a few attacks will drive the right-skewed degree distribution towards a more balanced, normal distribution, as we will show later in the results section.

We will now show that in a random failure scenario a right-skewed degree distribution will emerge due to the algorithm. To do so we will estimate $E[\Delta(deg(v))]$, i.e., the *expected change in the degree of* w. We assume that there is no correlation between the degree of neighbors, i.e., that the so-called *assortativity* of the graph is 0 [17] and thus the graph is *disassortative*. Newman has shown that the preferential attachment and the random graph model are disassortative. Neither A_i causes assortativity of the network if we start with a disassortativ graph. This implies that the degree of vertices in $N_2(v)$ cannot be correlated with the degree of v. Even if the new target vertex w is chosen in proportion to its degree (or any higher power of it) this does not introduce any assortativity since the degree of v is chosen uniformly at random from the set of all vertices. Thus, the graph stays disassortative, which will help us to estimate the number of vertices in the second neighborhood of any v.

We can now determine the probability of any vertex v to gain or lose a new edge in a random failure scenario by determining $E[\Delta(deg(v))]$, i.e. , by determining the rate equation of the degree. Let $P_-(v)$ denote the probability that v loses an edge by a random failure, and let $P_+(v)$ denote the probability that v gains an edge, then $E[\Delta(deg(v))] = P_+(v) - P_-(v)$. $P_-(v)$ is given by $0.5 \cdot deg(v)/n$ since there are $deg(w)/n$ ways to remove a direct neighbor of w and with probability 0.5 the removed edge will not be replaced and thus v loses an edge.

$P_+(v)$ is harder to determine since we have to change the perspective: first of all, one of the second neighbors of v has to lose its direct neighbor. The probability for this is

$$\sum_{z \in N_2(v)} \frac{deg(z)}{n} \tag{4}$$

If one of the vertices $z \in N_2(v)$ loses a neighbor, it will build a new edge with probability 0.5 and choose v from its second neighbors with probability $P_i(v, w)$.

Thus, $P_+(v)$ is given by:

$$P_+(v) = \sum_{z \in N_2(v)} 0.5 \frac{deg(z)}{n} \frac{deg(v)^i}{\sum_{z' \in N_2(z)} deg(z')^i} \quad (5)$$

Let now $deg_\emptyset$ denote the *average degree* in the graph, i.e., $deg_\emptyset = 2m/n$. Since we restrict i to be $\in \{0, 1\}$ we can approximate $\sum_{z' \in N_2(z)} deg(z')^i$ by

$$\sum_{z' \in N_2(z)} deg(z')^i \simeq deg(z) \cdot deg_\emptyset^{i+1} \quad (6)$$

With this we can simplify $P_+(v)$ to:

$$P_+(v) = \sum_{z \in N_2(v)} 0.5 \frac{deg(v)^i}{n \cdot deg_\emptyset^{i+1}} \quad (7)$$

$$= 0.5 \frac{|N_2(v)| deg(v)^i}{n \cdot deg_\emptyset^{i+1}} \quad (8)$$

$$= 0.5 \frac{deg(v)^{i+1}}{n \cdot deg_\emptyset^{i}}. \quad (9)$$

The last equation is again derived by the approximation $N_2(v) \simeq deg(v) \cdot deg_\emptyset$. Thus, for $i = 0$, the probability $P_+(v)$ to gain and the probability $P_-(v)$ to lose an edge are the same: $0.5\ deg(w)/n$. Thus, $E[\Delta(deg(w))] = 0$. This means that the degree of each vertex makes a random walk with the same probability to be increased or decreased [5]. Since the degree of a vertex cannot be below 0 this random walk is bounded from the left and will thus create a right-skewed degree distribution sooner or later. Note however that the right-shift will be very slow: after k steps of a normal random walk, the standard deviation is $\sqrt{k}$. Even if any vertex was the last to be deleted, the degree will thus expectedly not be higher than $\sqrt{n}$. But of course, every vertex will only perform a step on the random walk if either one of its direct neighbors is deleted, i.e., expectedly every $n/deg(v)$ deletions, or if one of the neighbors of its second neighbors is deleted, i.e., approximately after $n/(|N_2(v)| \cdot deg_\emptyset)$ deletions. Since both of these values are depending on the changing degree of v it is not easy to determine analytically how often any vertex will perform the random walk. But for fixed $deg_\emptyset$ and $\lim n \to \infty$ and with an upper bound on $deg(v) \sim \sqrt(n)$, the expected number of steps performed on the random walk is bound by $\sqrt(n) \cdot deg_\emptyset^3$ and thus the expected maximal degree is bound from above by $O(\sqrt{\sqrt{n}})$.

We will now examine algorithm A_1. For A_1 the approximated rate equation of the degree of v is given by:

$$E[\Delta(deg(v))] = 0.5 \frac{deg(v)}{n} \left(\frac{deg(v)}{deg_\emptyset} - 1 \right) \quad (10)$$

This is an interesting equation that shows that a vertex with a degree higher than average is likely to have a positive $E[\Delta(deg(v))]$ value, i.e., to gain new edges, while vertices with lower than average degree tend to lose edges. It is clear that this rule should easily be able to shift the degree distribution of the network to the right since it has an inbuilt self-enforcing mechanism: if a vertex' degree increases it has an even higher chance in the next step to gain more edges. We will show some experimental results of this behavior in the next section.

Note however that there are also some limitations of the self-enforcing mechanism:

1. The approximation of $|N_2(v)|$ by $deg(v) \cdot \deg_\emptyset$ is clearly not valid anymore if v has a degree of more than $n/deg_\emptyset$;

2. Since new edges emerge always between former second neighbors, the new edge forms a triangle. This implies that the clustering coefficient[1] increases strongly. Thus, many edges of direct neighbors of v do not lead to 'new' second vertices but to other direct neighbors of v. In this case, the above given approximation of $|N_2(v)|$ is likely to be an upper bound on most second neighborhoods;

3. Last but not least: after expectedly $n/2$ steps the vertex itself will be deleted due to some random failure.

In the following section we will show some empirical results for A_0 and A_1 and show which of the rules is likely to be applicable to a real-world scenario.

4. Results

To measure the robustness of the resulting networks we have simulated the model as described in Algo 2.

This experimental setting has to be discussed: first, we use a synchronized setting, i.e., vertices will rewire after each other. This is not necessarily how it would be implemented in a real system but it models a system where vertices are not immediately aware of a missing neighbor. Since everything is local, asynchronicity should not have a large effect. Second, we replace the deleted vertex z by a new vertex z' with half the edges of z, i.e., the number of vertices n is held constant. This

[1]The clustering coefficient of a given vertex v is defined as the ratio between the number of edges between its direct neighbors and the possible number $deg(v) * (deg(v) - 1)/2$ of these edges. The clustering coefficient of a graph is defined as the average clustering coefficients of its vertices [27].

Algorithm 2 Algorithm to test the rewiring procedure described in Algo 1.

```
1: procedure EXPERIMENTALREWIRING ▷
2:     Start with random graph G with n = 1000 and deg_∅ = 2.5;
3:     for k rounds do
4:         for 10,000 random failures and 10,000 attacks each do
5:             delete vertex z;
6:             for all neighbors v of z do
7:                 v.rewire();
8:             end for
9:             add new vertex z' by choosing deg(z)/2 neighbors at random
10:        end for
11:    end for
12: end procedure
```

is necessary since most structural measures are depending on a constant n, as $\mathcal{R}_A$ and $\mathcal{R}_{RF}$ or $E[\Delta(deg(v))]$. Note however, that the random connecting scheme of vertex z' will not introduce a right-skewed degree distribution by itself, i.e., all shifts in the degree distribution emerge by actions due to Algorithm 1. Furthermore, it keeps the total number of edges expectedly constant.

This reconnecting scheme increases the probability $P_+(v)$ of any non-deleted vertex v by $deg(z)/2n)$ which can be approximated by $deg_\emptyset/2n$ for a large number of deletions. Thus, Equ. 10 becomes:

$$E[\Delta(deg(v))] = 0.5\frac{deg(v)}{n}\left(\frac{deg(v)}{deg_\emptyset} - 1\right) + \frac{deg_\emptyset}{2n}. \tag{11}$$

Fig. 1 shows that Equ. 11 describes the observed average increase of the degree per deletion quite well. For this we simulated one round of $40,000$ random failures (uniformly at random) in a random graph with $n = 20,000$ and $deg_\emptyset = 5$. For degrees between 0 and 50, Equ. 11 describes the observation quite well, but after that range (not shown) it is only an upper bound on the observed values. This is due to the limited capabilities of the approximation for vertices with large degree as discussed above.

Fig. 2 shows that the variants A_0 and A_1 are indeed able to change a normal degree distribution (given by a random graph) to a right-skewed degree distribution under random failures. It is also clear to see that A_1 changes the degree distribution faster into a strongly right-skewed degree distribution, as expected by the theoretical analysis.

Note that neither A_0 nor A_1 can guarantee connectivity, i.e., it is possible that small subgraphs are disconnected from the biggest connected component. We thus measured the amount of vertices in the biggest connected component under A_0 and A_1: the percentage of vertices in the biggest connected component does not fall below 99% if A_0 is applied, and not below 98% for A_1 in the scenario shown in Fig. 2. Another interesting measure is the evolution of the average path length in this scenario: under A_0 the average path length increase by approx. 4%, while it decreases by the same amount under A_1. In summary, A_0 is slightly better in keeping the graph together, but the vertices in the biggest connected component are better connected under A_1.

Up to now we have only shown results on one round of random failures, starting from a random graph. Fig. 3 shows the resulting degree distributions of a graph that suffers 5 runs of attack and random failure series, starting with a scale-free graph or random graph with $n = 1,000$ and $m = 5,000$. Every series contains $1,000$ subsequent removals of each type. It is clearly visible that the degree distribution changes from a sharp degree distribution with a low deviation to a right-skewed distribution and vice versa, as predicted for both algorithms.

Of course, the degree distribution itself is just an indicator of the resulting network's robustness. Thus, we also measured $\mathcal{R}_A$ and $\mathcal{R}_{RF}$ every 100 deletions while the graph was attacked or suffered from random failure[2]. Note that these robustness measures are a bit unintuitive in that the *higher* the value, the *less robust* they are, since the larger the average path length after deleting 5% of the vertices the worse. Thus, we need to compare the values to those of pure random and pure scale–free graphs: a pure random graph with 1000 vertices and 5000 edges has a robustness $\mathcal{R}_{RF}$ against random failures of 3.3 and a robustness $\mathcal{R}_A$ of 3.4 against attacks. A scale–free graph constructed by the preferential attachment model of Barabási et al [3] and the same number of vertices and edges has a robustness against random failures of 3.0 and against attacks of only 3.5. I.e., as described by Albert and al. scale–free networks are most stable against random failures and least stable against attacks [1].

What is expected is that after some attacks, the robustness of the network against attacks should not be higher than 3.4 (as in the pure random graph model) and after some random failures, the robustness of the network against random failures should not be higher than 3.0 (as in the pure scale–free graph). As can be

[2]Note that to measure $\mathcal{R}_A$ and $\mathcal{R}_{RF}$ it is necessary to delete 5% of the vertices as defined in Section 2 to measure the increased average path length. Of course, these deleted vertices where only temporarily deleted and thereafter reinserted, i.e., the whole graph was restored before continuing with subsequent deletions.

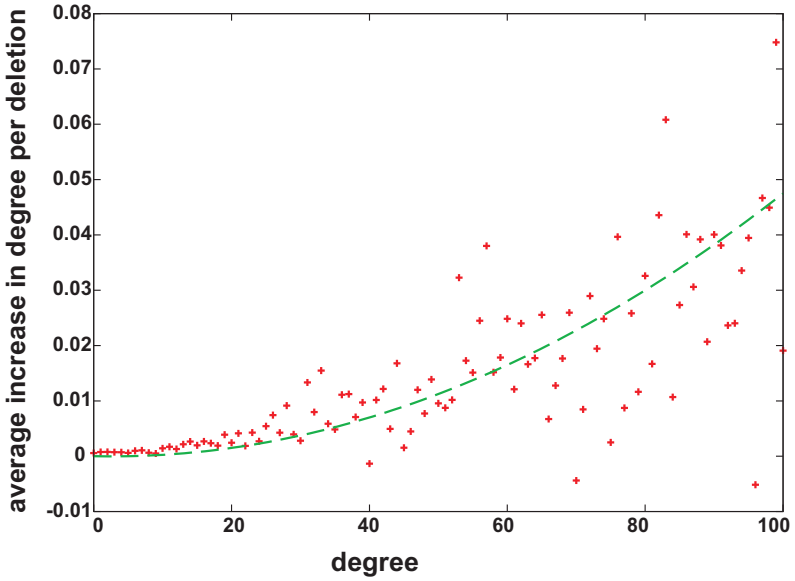


Figure 1. The figure shows the average increase of the degree per random failure. Denoted by a green, dotted line is the predicted value given by Equ. 11. Note that for larger degrees (not shown) Equ. 11 gives only an upper bound on the observed values.

seen in Fig. 3, both algorithms A_0 and A_1 quickly result in network structures that show the same robustness in the case of attacks as the scale-free graph, i.e., the produce graphs that are as sensible as a scale–free graph. Unfortunately, the emerging right-skewed degree distribution after the first 1000 random failures is not strong enough to decrease $\mathcal{R}_{RF}(G)$ to the wanted value of 3.0 but only to $\simeq 3.4$ in the case of A_0 and $\simeq 3.3$ in the case of A_1. This value is only comparable with the stability of a pure random graph but not with that of a pure scale-free graph. Of course, one can assume that normally random failure phases will be much longer than attack phases. Fig. 4 shows $\mathcal{R}_A(G)$ and $\mathcal{R}_{RF}(G)$ in the evolution of a random graph with $1,000$ vertices and $5,000$ edges suffering from $20,000$ random failures. It is clear to see that also a long random failure phase does not change the behavior of the network generated by A_0, and that the resulting network is less stable than a corresponding random graph with $\mathcal{R}_A(G) \simeq 3.6$ and $\mathcal{R}_{RF}(G) \simeq 3.4$. However, algorithm A_1 is able to generate a network whose random failure stability $\mathcal{R}_{RF}(G)$ fluctuates around the wanted value of3, with a minimum of 2.8.

In summary, A_0 is too weak to stabilize networks in a random failure scenario if they do not already have a right-skewed degree distribution. But both, the theoretical analysis and the empirical results show that application of A_1 stabilizes an attacked network fast. Furthermore, as long as the network suffers only random failures, the algorithm stabilizes or re-establishes a right-skewed degree distribution that stabilizes the network against random failures.

5. Related Work and Summary

There are two strings of work that follow a similar aim as this paper: papers of the first kind try to find a scale–free network structure that is at the same time robust against attacks and random failures. Paul et al. come to the conclusion that a pure scale–free graph is not likely to be robust in both scenarios [21]. They come up with a network structure that is highly structured and depends on a central vertex to which almost all other vertices are connected. Because of its strongly organized nature it is not likely that this network structure could be implemented by a decentrally working network generating algorithm. Other papers try to adapt a networks structure to, e.g., the load in the system [23]. Massoulié et al. build an overlay network structure that at the same time balances the load and tries to preserve the distances in the full network [14]. They focus on a network structure that balances the vertices degree since 'balancing degrees (...) improves the resilience of the system to link and node failures'. At least for the random failures of vertices, this assumption stands in contrast to the findings of

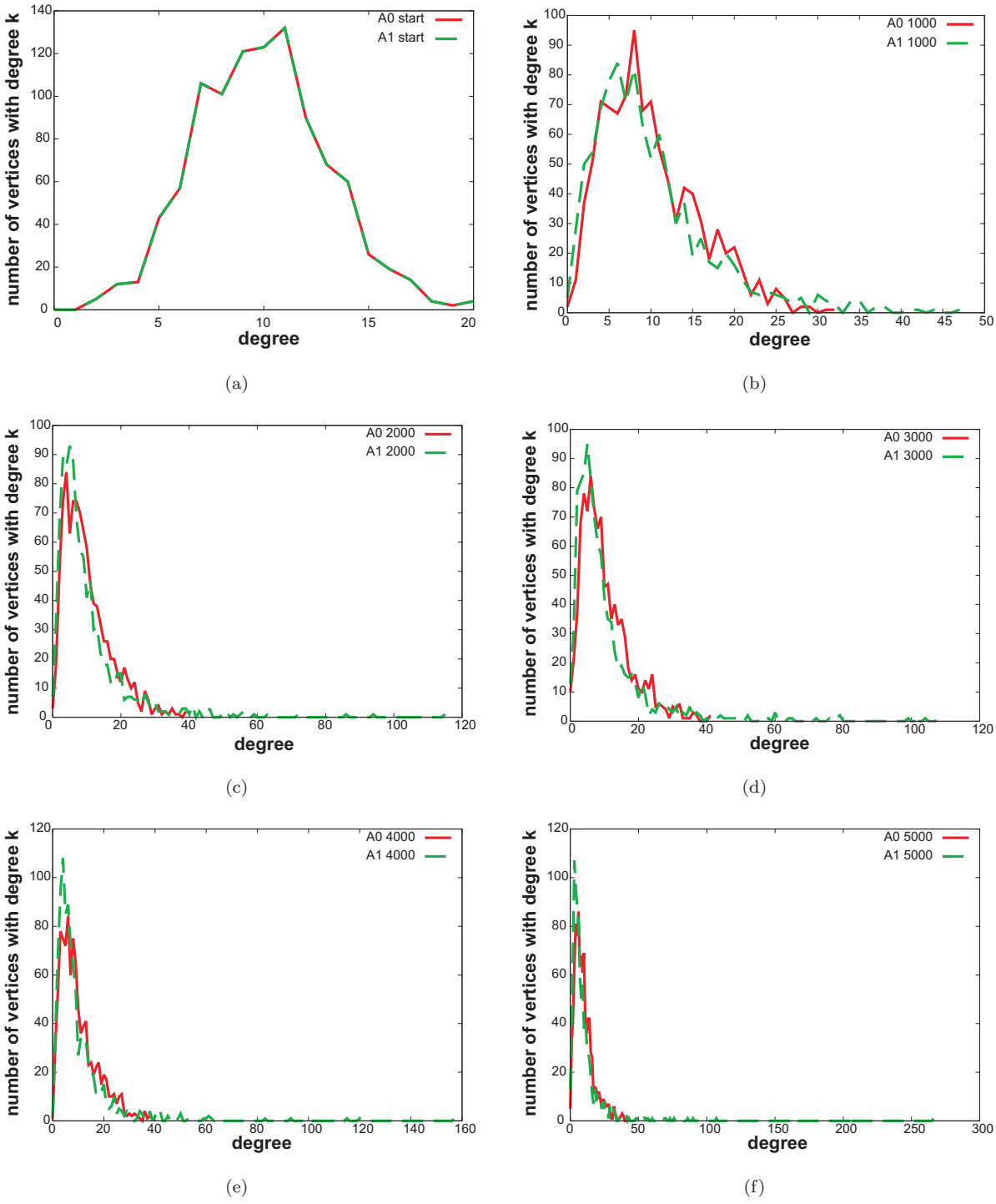
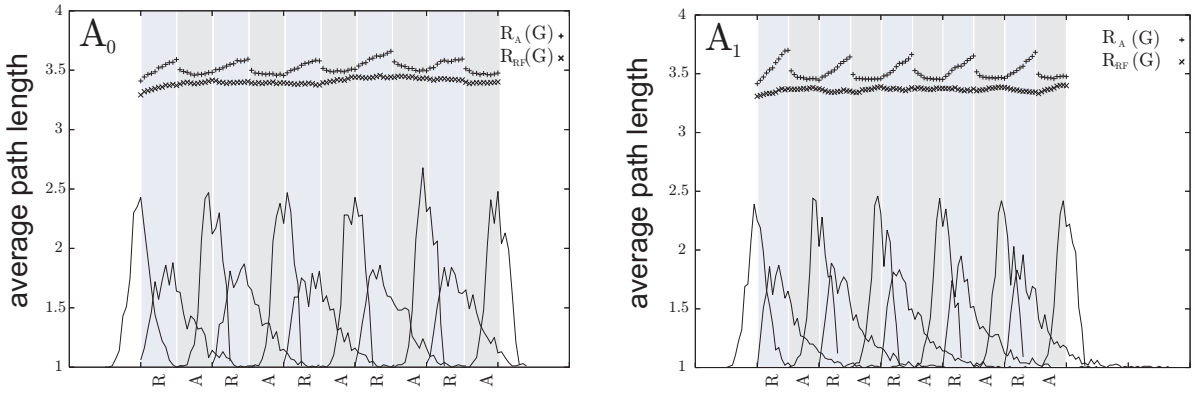


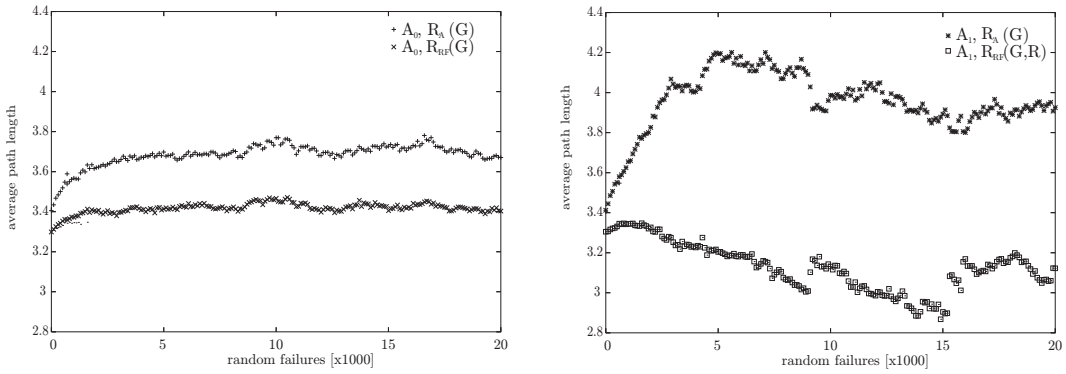
Figure 2. Exemplary evolution of the degree distribution of one random graph after $5,000$ random failures, plotted after every 1000 deletions. (a) Since both runs start with the same random graph with $n = 1000$ and $m = 5,000$, the degree distribution is the same for both algorithms. (b)-(f) Each diagram compares the resulting degree distribution after applying algorithm A_0 and A_1. It is clear to see that A_1 results in a degree distribution that is more right-skewed than the one created by A_0.



(a) starting with a random graph

(b) starting with a random graph

Figure 3. 5 **runs of** $1,000$ **random failures and** $1,000$ **each, indicated by the letters** A **and** R **and light and dark grey backgound color. We start with a random graph with** 1000 **nodes and** 5000 **edges and a run of random failures.The** y**-axis shows** $\mathcal{R}_A(G)$ **and** $\mathcal{R}_{RF}(G)$ **of the resulting graphs along the evolution. The degree distributions are shown out of scale to emphasize the correlation between a high** $\mathcal{R}_A(G)$ **value (litle robustness against attacks) and a right-skewed degree distribution. The first degree distribution represents the starting random graph, the second that after the first** $1,000$ **random failures, and the third the degree distribution after the next** $1,000$ **attacks, etc.**



(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

Figure 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** A_0 **and** A_1**. Starting graph is a random graph with** 1000 **vertices and** 5000 **vertices. (a) Algorithm** A_0 **creates a graph that is less robust against attacks than a pure random graph:** 3.6 **compared to** 3.4 **in a pure random graph. But the graph is still a bit more robust than a comparable scale–free graph with a value of** $\mathcal{R}_A(G) = 3.5$**. 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** A_1 **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.**

Albert et al. [1]. Their results concentrate on edge failures and they guarantee that the resulting graphs do not lose connectivity for a given failure rate of $(c-1)/c$. Keyani et al. also try to switch a network's structure in case of attacks: they implement a local attack detection that relies on a large fraction of missing neighbors, and they require each vertex to have a list of random vertices of the network [10]. As we have shown above, the switch to a narrow degree distribution is done by the nature of the attack itself, thus the other way, from narrow to right-skewed, is much more important.

As far as we know, this paper makes the first attempt to switch a network's structure in a decentrally and oblivious way as a reaction to attacks and random failures. It began as a simple theoretical gedanken-experiment, but now the question remains: can it be applied to real communication networks? It is surely debatable how much 'right–skewedness' of the degree distribution a real network can handle: in peer-to-peer systems the high degree vertices present super–users that need appropriate bandwidth and computing power that might not be available to all users. We have shown that for A_0 the maximal degree is bounded by $O(\sqrt{\sqrt{n}})$. An open question is whether the maximal degree under A_1 can be determined analytically. A possibility to control the maximal degree is to choose some non-integral value for i between 0 and 1 thus scaling between the two approaches A_0 and A_1. But it might also be that super–users are not so uncommon anymore in the near–future: with low–cost flatrates and cheap servers one could also think of rewarding another peer for accepting a connection request, either financially or by up–rating them. Since the rewarding peer has a benefit of the super–user such a system could be interesting for both sides. The other problem is of course to maintain other network structures that are, e.g., used for finding files or users. Since in the algorithm proposed here vertices mainly connect to vertices nearby it should be easy to maintain local informations. But since we are no experts in this field, we can just pose this as an open question. Another interesting aspect of the algorithm would be to analyze the dynamics of social networks in times of deadly and fast diseases, like the Spanish flu in the 1920s: since highly connected persons are most likely to get infected first this is some kind of attack on a social network that deprives it of its high degree vertices. It is known that social networks are at least to some degree scale–free [26, 2] and thus it would be very interesting to analyze the dynamics that recover this scale–free structure after the disease is cured.

One last finding implied by this article is that a scale-free network structure is only more vulnerable to attacks if the network is not allowed to react by building new edges to compensate for the ones that were deleted. We have shown that simple, local rules can be implemented that stabilize the network's structure at nearly no communication costs. Furthermore, since these rules are local, they stabilize the local network structure such that subsequent attacks will not enlarge the distance between near vertices but only those between far away vertices. Since in most communication networks local communication is much more probable than long-distance communication the newly introduced stability measure *coherence* shows that our network protocol stabilizes networks in a suitable way against attacks and random failures.

Acknowledgement

K.A.Z. is supported by a grant by the Deutsche Akademie der Naturforscher Leopoldina (BMBF LPD 9901/8-182).

References

[1] R. Albert, H. Jeong, and A.-L. Barabási. Error and attack tolerance of complex networks. *Nature*, 406:378–382, 2000.

[2] A.-L. Barabási. *Linked - The New Science of Network*. Perseus, Cambridge MA, 2002.

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

[4] B. Bollobás. *Random Graphs*. Cambridge Studies in Advanced Mathematics 73. Cambridge University Press, London, 2nd edition, 2001.

[5] P. Bremaud. *Markov Chains - Gibbs Field, Monte Carlo Simulation, and Queues*. Springer Verlag, 1999.

[6] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. *Computer Communications Review*, 29:251–262, 1999.

[7] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99:7821–7826, 2002.

[8] P. Holme. Congestion and centrality in traffic flow on complex networks. *Advances in Complex Systems*, 6:163, 2003.

[9] A. Iamnitchi, M. Ripeanu, and I. Foster. Small-world file-sharing communities. In *Proceedings of the IEEE INFOCOM 2004*, 2004.

[10] P. Keyani, B. Larson, and M. Senthil. *Web Engineering and Peer-to-Peer Computing: NETWORKING 2002*, chapter Peer Pressure: Distributed Recovery from Attacks in Peer-to-Peer Systems, pages 306–320. Springer Berlin/Heidelberg, 2002.

[11] J. Kleinberg. Navigation in a small world. *Nature*, 406:845, 2000.

[12] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, 2000.

[13] K. A. Lehmann and M. Kaufmann. *Peer-to-Peer Systems and Applications*, chapter Random Graphs, Small Worlds, and Scale-Free Networks. Springer Verlag, 2005.

[14] L. Massoulié, A.-M. Kermarrec, and A. J. Ganesh. Network awareness and failure resilience in self-organising overlay networks. In *Proceedings of the 22nd International Symposium on Reliable Distributed Systems (SRDS'03)*, 2003.

[15] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298:824–827, 2002.

[16] M. E. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences, USA*, 98(2):404–409, 2001.

[17] M. E. Newman. Assortative mixing in networks. *Physical Review Letters*, 89:208701, 2002.

[18] M. E. J. Newman and D. J. Watts. Scaling and percolation in the small-world network model. *Phys. Rev. E*, 60:7332–7342, 1999.

[19] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814, 2005.

[20] R. Pastor-Satorras and A. Vespignani. *Handbook of Graphs and Networks: From the Genome to the Internet*, chapter Epidemics and Immunization in Scale-Free Networks. Wiley-VCH, Berlin, 2002.

[21] G. Paul, T. Tanizawa, S. Havlin, and H. Stanley. Optimization of robustness of complex networks. *The European Physical Journal B - Condensed Matter and Complex Systems*, 38(4):187–191, 2004.

[22] M. Ripeanu and I. Foster. Mapping the gnutella network: Macroscopic properties of large-scale peer-to-peer networks. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems and Implications for System Design*, 2002.

[23] L. Rodero-Merino, A. F. Anta, L. Lopéz, and V. Cholvi. DANTE: A self-adapting peer-to-peer system. In *Proceedings of the Fifth International Workshop on Agents and Peer-to-Peer Computing (AP2PC 2006)*, pages 79–90, 2006.

[24] S. Saroiu, K. Gummadi, and S. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networks*, 2002.

[25] D. J. Watts. A simple model of global cascades on random networks. *Proceedings of the National Academy of Sciences*, 99(9):5766–5771, 2002.

[26] D. J. Watts. *Six Degreees - The Science of a Connected Age*. W.W. Norton & Company, New York, London, 2003.

[27] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.