The Markov Chain Simulation Method for Generating Connected Power Law Random Graphs[1]

CHRISTOS GKANTSIDIS[2]   MILENA MIHAIL[3]   ELLEN ZEGURA[4]

College of Computing
Georgia Institute of Technology

September 23, 2002

Abstract

Graph models for real-world complex networks such as the Internet, the WWW and biological networks are necessary for analytic and simulation-based studies of network protocols, algorithms, engineering and evolution. To date, all available data for such networks suggest heavy tailed statistics, most notably on the degrees of the underlying graphs. A practical way to generate network topologies that meet the observed data is the following degree-driven approach: First predict the degrees of the graph by extrapolation from the available data, and then construct a graph meeting the degree sequence and additional constraints, such as connectivity and randomness. Within the networking community, this is currently accepted as the most successful approach for modeling the inter-domain topology of the Internet.

In this paper we propose a Markov chain simulation approach for generating a random connected graph with a given degree sequence. We introduce a novel heuristic to speed-up the simulation of the Markov chain. We use metrics reminiscent of quality of service and congestion to evaluate the output graphs. We report experiments on degree sequences corresponding to real Internet topologies. All experimental results indicate that our method is efficient in practice, and superior to a previously used heuristic.

---

[2]Email: christos@cc.gatech.edu.
[3]Email: mihail@cc.gatech.edu.
[4]Email: ewz@cc.gatech.edu.

1 Introduction

There has been a recent surge of interest in complex real-world networks. These include *the WWW* [26, 35, 6, 9, 14, 27, 28] where a node corresponds to a Web page and there is an edge between two nodes if there is a hyperlink between the corresponding pages, the *Internet at the level of Autonomous Systems* (a.k.a. inter-domain level) [18, 25, 32, 36, 10, 11, 38, 5] where a node corresponds to a distinct routing administration domain (such as a University, a corporation, or an ISP) and an edge represents direct exchange of traffic between the corresponding domains, and *biological networks* [21] where, nodes correspond to genetic or metabolic building blocks (such as genes and proteins) and edges represent direct interactions between these blocks. Obtaining accurate graph models for such real-world networks is necessary for a variety of simulation-based studies.

A very robust and persistent characteristics of complex networks, including the WWW, the Internet and biological networks, is that, while the average degree is constant, there is no sharp concentration around the average degree and there are several vertices with very large degrees. Formally, the degree sequence follows heavy tailed statistics in the following sense: **(a)**The ith largest degree of the graph is proportional to $i^{-\alpha}$, with α approaching 1 from below, **(b)**The frequency of the ith smallest degree of the graph is proportional to $i^{-\beta}$, with β approaching 3 from below (see [18] for detailed Internet measurements, see [6, 14, 27, 28] for WWW measurements). This is a sharp departure from the Erdös-Rényi random graph model where the degrees are exponentially distributed around the mean. Consequently, several papers have proposed plausible graph models, based on the notion of "preferential attachment" [6, 8, 28, 4, 13] and on the notion of multiobjective optimization [17, 5] for explaining this phenomenon; the former is efficient to simulate but does not capture the primitives driving the evolution of the network, while the latter captures the evolutionary primitives directly, but is much slower to implement. Despite the elegant principles of the above approaches, none of them predicts accurately all the observed measurements. In fact, none of these approaches attempts to explain the heavy tailed statistics on the high-end and the low-end of the degrees, **(a)** and **(b)** above, *simultaneously*, and there is further evidence that **(a)** and **(b)** cannot be captured by a single evolutionary principle ([1] argues that a Pareto distribution should result in $\beta \simeq 1 + \frac{1}{\alpha}$, which is not the case for the observed numbers of the parameters α and β mentioned above). On the other hand, graph models for complex networks are often expected to pass strict performance requirements. For example, the networking community uses such graph models to simulate a wide range of network protocols [41, 18, 25, 31, 32, 36, 10, 11, 38, 5], and hence the accuracy of the underlying topology model is considered very important.

Therefore, the following alternative degree-driven approach for generating network topology models has been adopted. First predict the degrees of the graph to be generated by extrapolation from available data, for example, according to **(a)** and **(b)** above, and then generate a graph that satisfies the target degree sequence, and additional constraints, the first and most natural of which is connectivity. It has also been observed that connected graphs that satisfy the degree sequence and some further "randomness property" are good fits for real Internet topologies [38] (albeit, "randomness property" is not quantified in [38]).

In the theory community the above degree-driven approach was first formalized in [3, 12] who especially addressed the connectivity issue, by isolating ranges of the parameter β for which the

resulting random graph has a giant connected component. In particular, for target degree sequence $d_1 \geq d_2 \geq \ldots \geq d_n$ over vertices v_i, $1 \leq i \leq n$, [3] proposed to consider $D = \sum_i d_i$ vertices by expanding vertex v_i to d_i vertices, construct a random perfect matching of size $D/2$ over the D vertices, and consider a graph on the initial n vertices in the natural way: v_i is connected to v_j if and only if, in the random perfect matching, at least one of the d_i vertices that correspond to v_i is connected to one of the d_j vertices that correspond to v_j. [3] further proposed to eliminate self loops and parallel edges, and consider the largest component of the resulting graph. The advantages of this approach are its implementational efficiency, and the guarantee of uniform sampling. However, the approach also has two drawbacks: It does not produce a graph that matches the degree sequence exactly, and, for the critical range $2 < \beta < 3$, the method gives small components of size $\Theta(\log n)$ (while, for example, for $\beta < 2$ the small components would be of constant size). There is no known performance guarantee concerning how accurately the method of [3] approximates the target degree sequence.

In the networking community the same degree-driven approach is typified by the Inet topology generator [25], which is currently the method of choice. The implementation of Inet uses the following heuristic: It first predicts a degree sequence by using $d_i \simeq \alpha^{-1}$ for the highest 1% of the degrees, and frequency of the ith smallest degree proportional to $i^{-\beta}$ for the remaining 99% vertices. It then constructs a connected graph that meets a predicted degree sequence by placing a spanning tree to guarantee connectivity, and tries to match the remaining degrees "as much as possible" using a preferential connectivity heuristic. Again, there is no known performance guarantee on how well the method of [25] approximate the target degree sequence, or to what extend their graph approximates a graph sampled uniformly at random from the target degree sequence.

In this paper we propose a Markov chain simulation approach for generating a random connected graph with a given degree sequence. In Section 2 we review the necessary graph theory to obtain an initial connected realization of the degree sequence. We point out that the underlying theory allows great flexibility in the produced output. In Section 3 we point out a Markov chain on the state space of all connected realizations of the target degree sequence. We note that, even though similar Markov chains were considered before without the connectivity requirement, the additional connectivity requirement needs a non-trivial theorem of [39] to result in a connected state space. This Markov chain requires a connectivity test in every simulation step. In Section 4 we introduce a novel speed-up of the Markov chain which saves greatly on connectivity tests. For example, we can simulate 1M steps of the speed-up process in the same time as a few thousand steps of the original process. Section 5 contains experimental results. We use metrics reminiscent of quality of service and congestion to evaluate the output graphs. We report experiments on degree sequences corresponding to real Internet topologies. All experimental results indicate that our method is efficient in practice, and superior to a previously used heuristic.

2 Markov Chain Initialization: Erdös-Gallai Conditions and the Havel-Hakimi Algorithm

In this Section we address the problem of constructing a connected graph that satisfies a given target degree sequence, if such a graph exists. We point out that such constructions follow from

classical graph theory, and that they allow substantial flexibility in the generated output graph. We will use these constructions as initial states of the Markov chains of Sections 3 and 4. (In addition, these fundamental theoretical primitives can replace all ad-hoc heuristics of the current implementation of Inet[25]).

Let n denote the number of nodes of the graph we wish to generate. Let $v_i, 1 \leq i \leq n$ denote the nodes and $d_1 \geq d_2 \geq \ldots \geq d_n$ denote the intended degrees of these nodes. We would like a simple, undirected, connected graph meeting the above degree sequence. A sequence of degrees $d_1 \geq d_2 \geq \ldots \geq d_n$ is called *realizable* if and only if there exists a simple graph whose nodes have precisely this sequence of degrees. A straightforward necessary condition for a degree sequence to be realizable is that for each subset of the k highest degree nodes, the degrees of these nodes can be "absorbed" within the nodes and the outside degrees. Stated formally, for $1 \leq k \leq n-1$:

$$\sum_{i=1}^{k} d_i \leq k(k-1) + \sum_{i=k+1}^{n} \min\{k, d_i\}. \tag{1}$$

A necessary condition for the realization to be connected is that the graph contains a spanning tree, which means that:

$$\sum_{i=1}^{n} d_i \geq 2(n-1). \tag{2}$$

The Erdös-Gallai theorem states that these necessary conditions are also sufficient [7, 16]. The proof is inductive and provides the following construction, known as the Havel-Hakimi algorithm [19, 20]. The algorithm is iterative and maintains the residual degrees of vertices. In each iteration, it picks an arbitrary vertex v and adds edges from v to d_v vertices of *highest* residual degree, where d_v is the residual degree of v. The residual degrees of the latter d_v vertices are updated appropriately. The significance of connecting with d_v highest degree vertices is that it ensures that condition (1) holds for the residual problem instance.

For example, the algorithm can start by connecting the highest degree vertex v_1 with d_1 other high degree vertices and obtain a residual degree sequence by reducing the degrees of these vertices by one, and repeat the same process until all degrees are satisfied (otherwise output "not realizable"). Alternatively, the algorithm can connect the lowest degree vertex v_n with d_n (resp. or a randomly chosen vertex v_i) with the d_n (resp. d_i) highest degree vertices, reduce their degrees and proceed as above.

Clearly the above algorithm runs in n iterations, each iteration invoking the degree of a vertex (and some book-keeping for maintaining residual degrees in sorted order). Thus the running time is very efficient, both in theory and in practice. In addition, since the sequence in which it picks vertices can be chosen, it provides the flexibility alluded to above. For example, when we start with higher degree vertices we get topologies that have very "dense cores", while when we start with low degree vertices we get topologies that have very "sparse cores".[5]

---

[5] The Erdös-Gallai condition (1) allows for further flexibility, at the cost of additional tests for condition (1), and repeated efforts until condition (1) is satisfied. In particular, the d_v vertices can be chosen according to any criterion, provided that, after each iteration, we ensure that condition (1) is satisfied by the residual graph (this part was automatic in case maximum degree vertices are chosen). If not, the choice of the d_v vertices needs to be repeated. This observation indicates several ways in which the implementation of [25] can be improved, however, we shall refrain from such discussions since this is not the main focus of this paper.

Next, let us deal with the second requirement of obtaining a connected topology. If the graph constructed as described turns out to be unconnected, then one of the connected components must contain a cycle. Let (u, v) be any edge in a cycle and let (s, t) be an edge in a different connected component. Clearly, the graph does not have edges between the pairs u, s and v, t. By removing the edges (u, v) and (s, t), and inserting the edges (u, s) and (v, t), we merge these two components. Note that the resulting graph still satisfies the given degree sequence. Proceeding in this manner, we can get a connected topology.

3 A Markov Chain on Connected Graphs with Prescribed Degree Sequence

We now turn to the question of generating a *random* instance from the space of all possible connected graphs that realize a target degree sequence. In experiment, it has been observed that "random" such instances are good fits for several characteristics of complex network topologies [3, 38] (albeit, all these experiments fall short of guaranteeing that the generated instances are either "correct" connected realizations of the target degree sequence).

For any sequence of integers that has a connected realization, consider the following Markov chain. Let G_t be the graph at time t. With probability 0.5, G_{t+1} will be G_t (this is a standard trick to avoid periodicities). With probability 0.5, G_{t+1} is determined by the following experiment. Pick two edges at random, say (u, v) and (x, y) with distinct endpoints. If (u, x) and (v, y) are not edges then consider a graph G' by removing the edges (u, v) and (x, y) and inserting the edges (u, x) and (v, y). Observe that G' still satisfies the given degree sequence. We further have to check whether G' is a connected graph. If it is connected then we perform the switching operation and let G_{t+1} be G'. Otherwise we do not perform the switching operation and G_{t+1} remains G_t. It follows from a theorem of Taylor [7, 39] that, using the above switching operation, *any connected graph* can be transformed to *any other connected graph* satisfying the *same degree sequence* (we note that the proof of Taylor's theorem is somewhat more involved than the corresponding fact for realizations *without the connectivity constraint*; the latter fact is straightforward). It now follows from standard Markov chain theory [33, 37] that this Markov chain converges to a unique stationary distribution which is the uniform distribution over the state space of all connected realizations. This is because, by definition, all transitions have the same probability. Thus, in the limit, the above Markov chain will generate a random graph with the given degree sequence.

What is the mixing rate? Similar questions have been considered elsewhere [37, 22, 23, 24] without the connectivity requirement. In particular, it is known that uniform generation of a simple graph with a given degree sequence $\mathbf{d} = d_1 \geq d_2 \geq \ldots \geq d_n$ reduces to uniform generation of a perfect matching of the following graph $M_{\mathbf{d}}$ [29]: For each $1 \leq i \leq n$, $M_{\mathbf{d}}$ contains a complete bipartite graph $H_i = (L_i, R_i)$, where $|R_i| = n - 1$ and $|L_i| = n - 1 - d_i$. The vertices of R_i are labeled so that there is a label for each $1 \leq j \leq n$ other than i; let us denote these labels by $\{u_{i,1}, \ldots, u_{i,i-1}, u_{i,i+1}, \ldots, u_{i,n}\}$. In addition, for each $1 \leq i, j \leq n$ with $j \neq i$, $M_{\mathbf{d}}$ has an edge between $u_{i,j}$ and $u_{j,i}$. Now each perfect matching $\mathcal{M}$ of $M_{\mathbf{d}}$ gives rise to a unique realization G of $\mathbf{d}$ in the natural way: G has a link between v_i and v_j if and only if $\mathcal{M}$ contains the edge between $u_{i,j}$ and $u_{j,i}$. Similarly, each realization G of $\mathbf{d}$ is associated with $\prod_{i=1}^{n}(n-1-d_i)!$ perfect matchings

of $M_\mathbf{d}$. It is known that a random perfect matching $\mathcal{M}$, and hence a random realization G, can be generated in time polynomial in n, only when $\mathbf{d}$ corresponds to a *regular* or *near-regular* graph, or when $\mathbf{d}$ corresponds to a *bipartite* graph. This does not include the case of arbitrary power-law graphs, and hence that theory does not apply. Indeed, generating a random graph that meets an arbitrary degree sequence $\mathbf{d}$ is a major open problem, at least since the original paper of Jerrum and Sinclair on approximating permanents [22].

In addition, we note here that the problem of rapid mixing of *connected* realizations is strictly harder than that of arbitrary realizations without the connectivity requirement, as indicated by the following reduction: For a degree sequence $\mathbf{d} = d_1 \geq d_2 \geq \ldots \geq d_n$, introduce an additional vertex with degree n, thus forcing any realization of the new sequence to have the new vertex connected to every other vertex, and hence it is connected and the realizations are one-to-one.

We thus have to devise efficient heuristics to estimate the mixing rate of the Markov chain. We have used the following heuristic to decide if a particular run of the Markov chain has converged sufficiently: Consider one or more quantities of interest, and measure these quantities every T steps. For example, one such quantity could be the diameter. In Section 5 we will consider further quantities that are related to quality of service (and use average shortest path from a node to every other node as an indicator) and network congestion (and use number of shortest paths through a node, or link, as an indicator). We may use the criterion of the quantities having converged as an indication of convergence of the Markov chain. However, the quantities under consideration may not converge, even under uniform sampling. For example, the diameter appears to deviate substantially from its mean, in the sense that the variance appears large. Therefore, a better heuristic is to estimate (this method of sample averages has been first considered in [2]). In addition, we will consider two (or more) separate runs of the Markov chain, where the initial points of each run are qualitatively different. For example, we may consider a "dense core" and a "sparse core" starting point, as mentioned in Section 2. Now, we may consider the case where the sample averages converge to the same number for the two separate runs of the Markov chain.

4   Speed-Up of the Markov Chain Simulation

Notice that the main bottleneck in the implementation of the Markov chain of Section 3 is the connectivity test that needs to be repeated in every step. This connectivity test takes linear time, for example, using DFS. On the other hand, all other operations, namely picking two random edges and performing the swap take time $O(\log n)$ ($\log n$ for the random choice, and constant time for the swap).

In this Section we describe a process which maintains convergence to uniform distribution over all connected realizations, but, in practice, performs much fewer connectivity tests. In particular, we consider the following process. Initially we have a connected realization of the target degree sequence, as mentioned in Sections 2 and 3. Let us call this G_0. We will be also maintaining a *window W*. This will be an estimate of how many steps of the Markov chain we can simulate without a connectivity test, and still have a reasonable probability of having a connected realization G_W after W steps (however, we do not require that every intermediate step between G_0 and G_W is connected). Initially the window is $W = 1$. The algorithm proceeds in stages, each stage consisting

6

of W simulation steps without a connectivity test. In general, if after W simulation steps we ended in a connected realization, then we will accept this realization as the next state and we will increase the window for the next stage by one: $W = W + 1$. If after W simulation steps we ended in a non connected realization, then we will return to the connected realization of the beginning of the current stage and we will decrease the window of the next stage to half its current size: $W = \lceil W/2 \rceil$ (this heuristic was inspired by the linear increase, multiplicative decrease of the TCP protocol). In this way, we hope to result in much fewer connectivity tests.

Notice that the above process is not strictly a Markov chain. In fact, we need to argue that its stationary distribution remains the uniform distribution over the set of all connected realizations. To see this, realize that the above process can be partition in stages $P_1, P_2, \ldots, P_j, \ldots$, where P_j contains the transition from initial state at the beginning of stage j to final state at the end of stage j, for the window size W during stage j. Now notice that, for this stage, W is fixed, and the process P_j is a Markov chain on the state space of connected realizations. In addition, the process P_j is *symmetric*, in the sense that the probability of ending at state X given that we started at state Y is the same as the probability of ending at state Y given that we started at state X. This follows from the symmetry of the initial Markov chain, which holds with or without connectivity tests (all transitions, with or without connectivity tests, have the same probability). We may now invoke the well known fact that aperiodic symmetric Markov chains converge to the uniform distribution [33, 37], and conclude that each one of the processes P_j has a unique stationary distribution, which is uniform. Therefore, their synthesis $P_1, P_2, \ldots, P_j, \ldots$, also converges to the uniform distribution.

We will use the term *fast Markov chain*, or *Markov chain speed-up*, to refer to the process with the sliding window W (as we said, strictly speaking, this is not a Markov chain, but a synthesis of Markov chains).

5 Evaluation

In this Section we evaluate the efficiency of the Markov chain in Section 3 with the speed-up proposed in Section 4. Our main application focus is the case of Internet topologies at the level of Autonomous Systems. We use data made available by the National Laboratory of Applied Networking Research[34] as well as Traceroute at Oregon[30]. These involve sparse power-law graphs (average degree is less than 3), whose size (number of nodes) has gown from 3K in November 1997 (when the data collection started), to approximately 14K, today. The maximum size of these graphs, under the current numbering, will be 64K.

The main results are as follows:

(1) We observe that, all quantities related to (a)shortest paths from a node to every other node and (b)number of shortest paths through a node converge is less than 2M steps of the speed-up chain. The corresponding running time is less than 1K secs on a Pentium III.

(2) The convergence of the Markov chain without the speed-up, for the same running time, is substantially weaker. Thus the speed-up indeed resulted in improvement.

(3) The convergence time appears to scale slowly with the size of the topology, which is encouraging when larger topologies will need to be generated.

(4) The results of the real Internet, the method of the random matching [3] discussed in the Introduction, and the Markov chain, do not coincide in all parameters. This indicates that no model captures the real Internet in full accuracy, and so all models are plausible candidates. In particular, the Markov chain method has the advantage over the method of [3] that it achieves the exact target degree sequence (we will see that the degree sequence of [3] can differ from the target sequence substantially in the large degrees), while the method of [3] has the advantage over the Markov chain method that it samples from a provably uniform distribution.

(5) The Markov chain method has the following additional advantage. If we start from an extreme initial point, like a dense or sparse topology, we can measure the parameters of interest at intermediate simulation points, and see how these parameters converge. Such simulations can be useful in stress tests of protocols.

More specifically, we have measured the following quantities:

(1) For each node v, the *average path* from v is the average of the shortest path from v to every other node. In the networking context, this is an indication of the quality of service perceived by v. We will consider the quantities *mean average path*, when the mean is taken over all the nodes, as well as the *variance* and *maximum* of the average path.

(2) For each node v, the *maximum path* from v is the maximum length shortest path from v to every other node. In the networking context, this is an indication of the worst case quality of service perceived by v. Again, we will consider the *mean* and *maximum* over all nodes of the graph. Notice that the *maximum maximum path* is the diameter.

(3) For each link e, the *link load* of e is the number of shortest paths through e, when n^2 shortest paths from each node to every other node have been considered (we break ties at random). We normalize by dividing with n^2. In the networking context, this is an indication of the congestion of the link. We will consider the *mean*, *variance*, and *max* of the link load, over all links of the network.

In Tables 1 and 2 we indicate the convergence of the speed-up Markov chain for the degree sequence of Internet inter-domain topology on June 2002. Table 1 corresponds to a dense initial topology, and Table 2 corresponds to a sparse initial topology. The last column correspond to a Markov chain without the speed-up for the same number of connectivity tests (thus approximately the same running time). We may observe, both the convergence of the speed-up Markov chain, as well as the weaker behavior of the slow Markov chain: For example, look at the Max Average Path and the Max Max Path (diameter) of the slow Markov chain. We have repeated these experiments 10 times, and the results are almost identical.

In Table 3 we compare the output of the Markov chain method to those of the power-law random graph (PLRG) method of [3], and to the results of the real Internet topology. It is clear that the Markov chain is a better fit. In addition, in experiment, we have found that the PLRG method of [3] produces graphs whose highest degrees deviate substantially from the real Internet topology.

In Tables 4 and 5 we indicate how the converge time, as the topology scaled, from approximately 3K nodes to approximately 14K nodes. We consider the two metrics of average path and normalized average link load. We should note that the number of steps indicated is the minimum number of steps that we needed to take in order for the metrics under consideration to converge accurately. The first thing to notice is the robustness of the method. We have actually performed further

Property	Initial	Fast MC 0.5M steps	Fast MC 1M steps	Fast MC 2M steps	Slow MC
Mean Average Path	13.9920	3.4277	3.4255	3.4252	3.4671
Variance Average Path	601.5923	0.2811	0.2795	0.2794	0.3044
Max Average Path	200.6095	7.9392	7.9760	8.0868	7.4373
Mean Max Path	205.8013	7.9634	7.9992	8.1179	7.4931
Variance Max Path	1.2605	0.4089	0.4070	0.3999	0.4184
Max Max Path (diameter)	213	12	12.1	12.3	11
Mean Link Load	4.94e-4	1.21e-4	1.20e-4	1.20e-4	1.22e-4
Variance Link Load	1.20e-5	3.17e-7	3.18e-7	3.18e-7	2.90e-7
Max Link Load	7.97e-2	5.49e-2	5.53e-2	5.55e-2	5.41e-2
Connectivity Tests		14,257	27,878	55,535	27,878
Running time (sec)		222.17	434.18	887.59	411.67

Table 1:

Property	Initial	Fast MC 0.2M steps	Fast MC 0.5M steps	Fast MC 1M steps	Slow MC
Mean Average Path	4.6661	3.4296	3.4270	3.4282	3.2726
Variance Average Path	0.8181	0.2837	0.2802	0.2826	0.1977
Max Average Path	10.1008	8.0073	8.0158	8.2066	6.5994
Mean Max Path	10.1693	8.0307	8.0451	8.2490	6.8558
Variance Max Path	1.3050	0.4101	0.4141	0.4013	0.3355
Max Max Path (diameter)	14	12	12	12.3	10
Mean Link Load	1.64e-4	1.21e-4	1.21e-4	1.21e-4	1.16e-4
Variance Link Load	1.44e-6	3.28e-7	3.26e-7	3.26e-7	2.55e-7
Max Link Load	5.95e-2	5.68e-2	5.64e-2	5.65e-2	4.50e-2
Connectivity Tests		4,165	12,224	25,736	12,224
Running time (sec)		64.82	190.11	397.47	176.20

Table 2:

simulations on parts of the network (such as Europe, or North America), and on projected degree sequences for the network in the next 5 years (as given by the Inet topology generator[25], and we have found qualitatively similar results. The second important thing to notice is that the scaling of the convergence time from the 3K node topology to the 13K node topology is very mild. We find this an encouraging evidence, that the convergence times will be reasonably efficient, as the topologies scale further.

6   Further Directions

A very challenging open problem in this area is the following: In addition to a target degree sequence, we may consider an underlying metric of distances. (In reality, these would be geographic distances for which data is available). We would then want to construct a minimum cost connected realization of the degree sequence. Even without the connectivity requirement, for a degree sequence on n nodes, this would be a mincost perfect matching problem on $O(n^2)$ nodes (along the reduction

Property	Dense		Sparse		Internet	PLRG
	Initial	Final	Initial	Final		
Path mean	13.9920	3.4261	4.6661	3.4337	3.6316	3.8735
Path var	601.5923	0.2851	0.8181	0.2863	0.3181	0.4738
Path max	200.6095	8.2758	10.1008	8.3023	7.5005	10.2855
Load mean	4.94e-4	1.21e-4	1.65e-4	1.21e-4	1.28e-4	1.54e-4
Load var	1.20e-5	3.17e-7	1.44e-6	3.30e-7	2.94e-7	1.21e-7
Load max	7.97e-2	5.58e-2	5.95e-2	5.67e-2	5.26e-2	1.94e-2

Table 3:

Graph	Nodes	Links	Internet	PLRG	Dense			Sparse		
					Initial	Steps	Final	Initial	Steps	Final
1997	3055	5678	3.7676	3.9802	4.0489	1000000	3.5744	5.2917	1000000	3.6046
1998	3666	7229	3.7376	3.9009	4.0732	1100000	3.5335	5.3540	800000	3.5554
1999	5053	10458	3.7148	3.9410	5.0164	900000	3.5270	5.5231	800000	3.5353
2000	7580	16153	3.6546	3.9020	5.7000	600000	3.4717	5.8082	1300000	3.4884
2001	10915	22621	3.6350	3.8422	7.0962	800000	3.4107	4.8178	900000	3.4169
2002	13155	27041	3.6316	3.8735	6.1559	1100000	3.4255	4.6610	1000000	3.4282

Table 4:

Graph	Nodes	Links	Internet	PLRG	Dense			Sparse		
					Initial	Steps	Final	Initial	Steps	Final
1997	3055	5678	7.19e-4	8.62e-4	1.01e-1	1000000	6.82e-2	1.00e-3	1000000	6.83e-2
1998	3666	7229	5.62e-4	6.57e-4	8.56e-2	1100000	5.66e-2	7.01e-4	800000	5.65e-2
1999	5053	10458	3.85e-4	4.54e-4	9.92e-2	900000	3.66e-2	3.69e-4	800000	3.68e-2
2000	7580	16153	2.42e-4	2.88e-4	8.50e-2	600000	2.30e-2	3.84e-4	1300000	2.32e-2
2001	10915	22621	1.61e-4	1.91e-4	6.23e-2	800000	1.15e-2	7.85e-5	900000	1.16e-2
2002	13155	27041	1.28e-4	1.54e-4	4.94e-4	1000000	1.21e-4	1.65e-4	1000000	1.21e-4

Table 5:

of Section 3). For n equal to several tens of thousands, all known exact mincost perfect matching algorithms are inefficient. Is there an efficient approximation [40]?

References

[1] Adamic, L., "Zipf, Power-Laws, and Pareto, a Ranking Tutorial". http://www.hpl.hp.com/shl/papers/ranking/, 2000.

[2] Aldous, D., and Fill, J., "Reversible Markov Chains and Random Walks on Graphs", Monograph, http://stat-www.berkeley.edu/users/aldous/book.html.

[3] Aiello, W., Chung, F.R.K. and Lu, L., "A Random Graph Model for Power Law Graphs", Proceedings of the Fourty-First Annual IEEE *Symposium on Foundations of Computer Science*, (FOCS 2000), pp. 171-180.

[4] Aiello, W., Chung, F.R.K. and Lu, L., "Random Evolution in Massive Graphs", Proceedings of the Fourty-Second Annual IEEE *Symposium on Foundations of Computer Science*, (FOCS 2001), pp. 510-519.

[5] Alderson, D., Doyle, J., Willinger, W., "Toward an Optimization-Driven Framework for Designing and Generating Realistic Internet Topologies", HotNets 2002.

[6] Barabási, A.-L. and Albert, R., "Emergence of scaling in random graphs", Science 286 (1999), pp. 509-512.

[7] Berge, Claude, "Graphs and Hypergraphs", North Holland Publishing Company, 1973.

[8] Bollobás, B., Riordan, O., Spencer, J. and Tusnády, G., "The degree sequence of a scale-free random graph process", Random Structures and Algorithms, Volume 18, Issue 3, 2001, pp 279-290.

[9] Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomikns, A. and Wiener, J., "Graph structure in the Web", Proc. 9th International World Wide Web Conference (WWW9)/Computer Networks, 33(1-6), 2000, pp. 309-320.

[10] Bu., T., and Towsley, D., "On Distinguishing between Internet Power Law Topology Generator", Infocom 2002.

[11] Chen, Q., Chang, H., Govindan, R., Jamin, S., Shenker, S., and Willinger, W., "The Origins of Power-Laws in Internet Topologies Revisited", Infocomm 2002.

[12] Chung, F.R.K. and Lu, L., "Connected components in random graphs with given degree sequences", Available at http://www.math.ucsd.edu/ fan.

[13] Cooper, C. and Frieze, A., A general model for web graphs, *Proceedings of ESA*, 2001, pp 500-511.

[14] Dill, S., Kumar, R., McCurley, K., Rajagopalan, S., Sivakumar, D. and Tomkins, A., "Self-similarity in the Web", In Proceedings of International Conference on Very Large Data Bases, Rome, 2001, pp. 69-78.

[15] Dorogovtsev, S.N. and Mendes, J.F.F., "Evolution of Networks", *Advances in Physics*, to appear (2002). Available at http://www.fc.up.pt/fis/sdorogov.

[16] Erdös. P. and Gallai, T., "Graphs with Prescribed Degrees of Vertices", *Mat. Lapok, 11*, 1960.

[17] Fabrikant, A., Koutsoupias, E. and Papadimitriou, C.H. "Heuristically Optimized Tradeoffs", Available at http://www.cs.berkeley.edu/ christos.

[18] Faloutsos, M., Faloutsos, P. and Faloutsos, C., "On Power-law Relationships of the Internet Topology", In Proceedings *Sigcomm* 1999, pp 251-262.

[19] Hakimi, S.L., "On the Realizability of a Set of Integers as Degrees of the Vertices of a Graph", *SIAM J. Appl. Math. 10*, 1962.

[20] Havel, V., "A Remark on the Existence of Finite Graphs", *Caposis Pest. Mat. 80*, 1955.

[21] Jeong, H., Tombor, B., Albert, R., Oltvai, Z., Barabasi, A., The large-scale Organization of Metabolic Networks, Nature 407, 651 (2000).

[22] Jerrum, M., and Sinclair, A., "Approximating the Permanent", SIAM J. of Computing 18, 1989, pp 1149-1178.

[23] Jerrum, M., and Sinclair, A., "Fast Uniform generation of Regural Graphs", TCS 73 (1990), pp 91-100.

[24] Jerrum, M., Sinclair, A., and Vigoda, E., "A polynomial time approximation algorithm for the parmanent of a matrix with non-negative entries", STOC 2001, pp 712-721.

[25] Jin, C., Chen, Q. and Jamin, S., "Inet: Internet Topology Generator", University of Michigan technical Report, CSE-TR-433-00. Available at http://irl.eecs.umich.edu/jamin.

[26] Kleinberg, J., "Authoritative sources in a hyperlinked environment", Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998. Extended version in Journal of the ACM 46(1999).

[27] Kumar, R., Rajagopalan, S., Sivakumar, D. and Tomkins, A., "Trawling the web for emerging cyber-communities", *WWW8/Computer Networks*, Vol. 31, No 11-16, 1999, pp. 1481-1493.

[28] Kumar, R., Raghavan, P., Rajagopalan, S., Sivakumar, D., Tomkins, A. and Upfal, E., "Stochastic models for the Web graph", *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*, (FOCS 2000), pp 57-65.

[29] Lovász, L., and Plummer, M.D., *Matching Theory*, Academic Press, New York, 1986.

[30] Public Route Server and Looking Glass Site List, Traceroute.org, http://www.traceroute.org.

[31] Medina, A., Lakhina, A., Matta, I. and Byers, J., BRITE: Universal Topology Generation from a User's Perspective. Technical Report BUCS-TR2001 -003, Boston University, 2001. Available at http://www.cs.bu.edu/brite/publications.

[32] Medina, A., Matta, I. and Byers, J., "On the origin of power laws in Internet topologies", *ACM Computer Communication Review*, vol. 30, no. 2, pp. 18–28, Apr. 2000.

[33] Motwani, R. and Raghavan, P., *Approximation Algorithms*, Cambridge University Press, 1995.
[34] National Laboratory for Applied Networking Research, Routing Data, http://moat.lanr.net/Routing/rawdata.
[35] Page, L., Brin, S., Motwani, R., Winograd, T., "The PageRank Citation Ranking: Bringing Order to the Web", Stanford Digital Library Technologies Project, 1998.
[36] Palmer, C. and Steffan, J., "Generating network topologies that obey power laws", In Proceedings of *Globecom* 2000.
[37] Sinclair, A., "Algorithms for Random Generation and Counting: A Markov Chain Approach", Springer-Verlag, 1997.
[38] Tagmunarunkit, H., Govindan, R., Jamin, S., Shenker, S., Willinger, W. "Network Topology Generators: Degree-Based vs Structural", Sigcom 2002.
[39] Taylor, R., "Constrained Switchings in Graphs", *SIAM J. ALG. DISC. METH.*, Vol. 3, No 1, 1982, pp 115-121.
[40] Vazirani, V., *Approximation Algorithms*, Springer-Verlag, 2001.
[41] Zegura, E., Calvert, K., Donahoo, M., "How to model an Internetwork", Infocomm 1996.