Generating Network Models Using the S-Metric

Isabel Beichl and Brian Cloteaux
Mathematical and Computational Sciences Division
National Institute of Standards and Technology
Gaithersburg, Maryland, U.S.A.

Abstract ***The ability to create random models of real networks is useful for understanding the interactions in these systems. Several researchers have proposed modeling complex networks by using the node degree distribution, the most popular being a power-law distribution. Recent work by Li et al. introduced the S metric as a metric to characterize the structure of networks with power-law distributions. In this paper, we examine some of the practical difficulties of producing random graphs with a given degree sequence and an approximate S value. We give a solution for this problem that we have had success using in our research.***

Keywords: network models, S metric, threshold acceptance

1 Introduction

A type of data of increasing importance in various areas of research is one that is based not on floating point values but rather on relationships between objects. These associations can be modeled as graphs or networks. Being able to create realistic network models is necessary for understanding the interactions involved in these systems. In this paper, we describe a method we have successfully used for creating network models constrained by an S metric value.

Many real world systems cannot be approximated using simple random graphs (or more precisely, the Erdős-Rényi random graph model). This is because the degree distribution of these random graphs follows a Poisson distribution. For many complex systems, however, it has been shown [1] that the degree distribution of the resulting networks more closely follows a power law distribution. In other words, the probability that a node has k adjacent edges is $P(k) \sim k^{-\alpha}$ for some $\alpha > 1$. This distribution in a network produces a few nodes with high degree (often called *hub nodes*) and a large number of low degree nodes.

The importance of networks that have an approximate power law distribution lies in the number of application areas in which they are found. These networks have been shown to arise naturally in systems of both biological [2, 3, 4] and social [5] interactions. They also appear in many engineered systems such as the power grid [2], the Internet [6], and software components [7].

Nevertheless generating random graphs with power law distributions is still not sufficient to model real networks. Informally, the reason for this is that there can be a large number of non-isomorphic graphs that share a particular degree sequence. Thus we can see a large variability in the characteristics of the graphs that share a common degree sequence.

Recent research has been looking at the structure of these networks to distinguish among them. In a paper by Li et al. [8], the authors introduce the S metric as a structural metric for characterizing this connectedness of the high degree nodes in networks having a power law distribution. The advantages of this metric is that a non-normalized version is simple to compute and it has been shown to be able to distinguish between many graphs with identical degree sequences. Also, it has been shown that using this metric to create models of networks tends to produce graphs with better structural characteristics than using only a power law distribution [9].

This purpose of this paper is to explore some of the practical issues involved with generating random network models with a given S value. We first define the S metric. Then we will describe a technique we have used to generate graphs with approximate S values, and finally we discuss an approximation scheme for quickly computing the s_{max} value for degree sequence.

2 The S metric

The basic concept of the S metric is to measure how interconnected the nodes of high degree (or hub nodes) are to each other in a graph. The definition of the S metric by Li et al. [8] is actually a normalized version of another metric that they call the s metric. Before we can give a definition of the s metric , we first must define some notation that we will use. A graph $G = (N, E)$ has a node set N and an edge set E. The degree sequence of G is $\omega = \{\omega_1, \omega_2, ..., \omega_{|N|}\}$ where the degree of $n_i \in N$ is ω_i. To show that an edge set E (or node set N) belongs to a graph G, we write $E(G)$ (or $N(G)$). To show that a degree sequence ω belongs to a graph G, we write $\omega(G)$. The set of all graphs with the degree sequence ω is represented as $\mathcal{G}(\omega)$. The definition of the s metric for a graph G is

$$s(G) = \sum_{(i,j) \in E(G)} \omega_i \cdot \omega_j \qquad (1)$$

The S metric introduces a normalization factor to the s metric. This is defined by

$$S(G) = \frac{s(G)}{s_{max}\left(\omega(G)\right)} \qquad (2)$$

where $s_{max}(\omega) = \max\{s(G) | G \in \mathcal{G}(\omega)\}$. This normalization factor allows us to compare networks of differing sizes. Otherwise, the s metric can only be profitably used to compare networks with the same degree sequence.

An investigation of the S metric has shown that it is able to distinguish between many graphs having the same degree sequence [8], and basing random network construction on it can give better structural characteristics [9]. One problem with using the S metric is that while it is trivial to compute the s metric for a graph, it can be slow and difficult to find s_{max} for computing S. We will examine a simple approximation scheme for s_{max} that also produces error bounds in section 4. First, we examine how to generate a random graph with an approximate s value.

3 Generating graphs with an approximate s value

When we talk about generating a graph with a given degree sequence ω and s value s_p, we are implicitly saying that the graph has a s value of $s(G) \in [s_p - \epsilon, s_p + \epsilon]$ where $\epsilon > 0$. For the given s value s_p and degree sequence ω, there is no guarantee that a graph G exists that meets those constraints. Thus, the parameters we expect for constructing a graph are ω, s_p, and ϵ.

In order to generate a model with an approximate s value, we use a random walk over the space of connected graphs with identical degree sequences. Two graph G_1, G_2 are adjacent in this space if there exist the edges $(u, v), (x, y) \in E(G_1)$ and $(u, x), (v, y) \in E(G_2)$. In other words, we can transform the current graph to a new graph with the same degree sequence using the degree-preserving switch defined by

$$[(u, v), (x, y)] \rightarrow [(u, x), (v, y)] \qquad (3)$$

This degree switch is shown in figure 1.

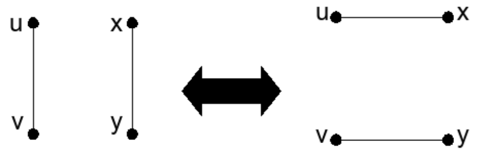

Figure 1: The degree preserving switch $[(u, v), (x, y)] \rightarrow [(u, x), (v, y)]$

We minimize the difference between the s value of the current graph and the desired value by picking switches using a form of simulated annealing called threshold acceptance [10]. Threshold acceptance is an optimization technique that accepts any transition in the state space that does not increase the overall difference between the s value of the current graph and s_p by more than a defined threshold. Thus for any randomly chosen switch of two edges in a graph, we would accept the switch to produce a new graph G' if $|s(G') - s_p| < |s(G) - s_p| + t$. This threshold is non-negative and decreases to zero in time. We return the graph G' if $|s(G') - s_p| \leq \epsilon$.

A threshold function depends on the maximum allowable change in s value that one switch can cause. An examination shows that this maximum change would occur if we had two edges that connected nodes with degree values ω_{max} and one where ω_{max} is the maximum value in the degree sequence ω. Thus the maximum change is $\omega_{max}^2 - O(\omega_{max})$. Using this, we define the value of our threshold function at time n to be $t(n) = p(n) \cdot \omega_{max}^2$ where $p(n) \in [0, 1]$ specifies a percentage of the maximum switch for the threshold. In practice, we typically start p from 0.1 and linearly decrease it to 0 (see [10] for a discussion on rate schedules).

The major difficulty in this random walk approach is that we need to maintain the additional constraint that the graphs must also be connected.

Any given switch can potentially disconnect the graph. In general, checking for connectivity is by far the most computationally expensive part of walking the graph space. Researchers have faced the same problem for sampling graphs with a given degree sequence using a Markov chain. We use the established idea of batching up a set of switches and testing if the graph is connected at the end of all the switches. If the resulting graph is connected, then the algorithm continues. Otherwise, the algorithm backtracks to the point of the last check and reduces the size of the number of switches in the batch before the next connectivity check. There has been recent research into finding optimal schemes to perform this type of connectivity checking [11, 12].

One advantage we have in generating graphs instead of performing sampling is that we do not have to consider all possible walks. In particular, for any two edges, if the degree of any two of the edge nodes is one, then we reject this switch automatically. The reason why we can reject this switch is that there are only two possible outcomes: if both nodes of degree one are matched to each other, then the resulting graph becomes disconnected and if the one degree nodes are not matched to each other, then the resulting graph is isomorphic to the original graph and so the s value does not change. Since power law distributions produce a large number of nodes with degree one, the most common type of switch to cause a disconnect in a graph is by matching two nodes of degree one. By ensuring that this case can never happen, we greatly reduce the chances of a switch disconnecting the graph.

Although all possible networks have a nonzero probability of being generated using this technique, we do not know the sampling distribution of these graphs. By extending the basic ideas in this method, it might be possible to to create a uniform sampling algorithm by using the Markov chain created by the space of connected graphs with s values between $[s_p - \epsilon, s_p + \epsilon]$. Starting from the graph we generated using threshold acceptance, we can then walk the Markov chain until we reach its stationary distribution. The difficulty with this technique is to show that the chain for a given ϵ is irreducible. In other words, every possible graph with an s value in $[s_p - \epsilon, s_p + \epsilon]$ is reachable with a sequence of s value preserving switches from any other graph with an s value in the correct range. If the Markov chain for a given ϵ does prove to be irreducible, it is simple to extend that chain to be ergodic. There is empirical evidence that for random walks over the set of connected graphs with degree ω, if the graphs have m edges, then it requires $O(m)$ edge swaps to reach the stationary distribution of the chain [12]. This result could suggest that a fast uniform sampling algorithm exists for the set of graphs with a degree sequence ω and an approximate s value. As excellent overview of issues involved in sampling using Markov chains is given by Randall [13].

4 Computing s_{max}

An algorithm for finding a graph with a given s_{max} value was given by Li et al. [8]. That algorithm is a greedy heuristic that passes through all the potential edges for graphs with the degree sequence and chooses the one that adds the most value to the s value of graph at each step. The algorithm is called a heuristic since there is no proof that the algorithm is able to complete in all cases. It was shown however that if the algorithm completes, then the resulting graph G must have an s value of $s_{max}(\omega(G))$.

There are three major drawbacks of the Li et al. algorithm. The first is that for larger degree sequences it is extremely slow. In our experience, using this method for computing s_{max} in networks over 10,000 nodes starts to become impractical. A second problem, that is related to the first, is that the algorithm is also memory intensive since it must hold a number of potential edges. Finally, the algorithm is complicated to implement and to optimize. Fortunately, for our purposes in generating graphs, we do not need the exact value of s_{max}. A good approximation is sufficient.

In order to approximate s_{max}, we modify an algorithm of Blitzstein and Diaconis [14]. The original algorithm was designed to create random graphs with a given degree sequence. Instead of random connections between the nodes in the graph, we instead choose the edge that connects the two highest degree nodes and still allows the remaining sequence to be realizable in a graph (or *graphical*). Before stating the algorithm, we need to first define some notation that we borrow from the Blitzstein and Diaconis paper [14].

If we have a degree sequence $\omega = \{\omega_1, \omega_2, ..., \omega_n\}$ and have the distinct indices $i_1, ..., i_k \in \{1, ..., n\}$, then the $\ominus_{i_1,...,i_k}\omega$ operator produces a new degree sequence where the values at the indices $\{i_1, ..., i_k\}$ are decreased by one and the other values remain the same. More formally,

$$(\ominus_{i_1,...,i_k}\omega)_i = \begin{cases} \omega_i - 1 & \text{for } i \in \{i_1, ..., i_k\} \\ \omega_i & \text{otherwise} \end{cases} \quad (4)$$

An example of this operator is for the degree sequence $\omega = (5,3,2,2)$, then $\ominus_{2,4}\omega = (5,2,2,1)$. Using this notation, then we state our approximation scheme in algorithm 1.

Algorithm 1 Creates a graph whose s value approximates s_{max}

```
Require: a graphical degree sequence $\omega = \{\omega_1, \omega_2, ..., \omega_n\}$ where $\forall i, j$ if $i < j$ then $\omega_i > \omega_j$
  $E \leftarrow \phi$
  $n_1 \leftarrow 1$
  $n_2 \leftarrow n_1 + 1$
  while $\exists i$ such that $\omega_i > 0$ do
    while $\omega_{n_1} > 0$ do
      if $\omega_{n_2} > 0$ and $\ominus_{n_1,n_2}\omega$ is graphical then
        $E \leftarrow E \cup (n_1, n_2)$
        $\omega \leftarrow \ominus_{n_1,n_2}\omega$
      end if
      $n_2 \leftarrow n_2 + 1$
    end while
    $n_1 \leftarrow n_1 + 1$
    $n_2 \leftarrow n_1 + 1$
  end while
  return $E$
```

The algorithm sequentially loops over the index with the highest degree that has not been matched with other nodes. This looping requires $O(|\omega|^2)$ iterations. The only other part of the algorithm that needs to be accounted for is to check if a degree sequence ω is graphical. Using the Erdős-Gallai theorem, a sequence can be checked in time linear to the number of nonzero degrees it contains. There are also established ways to reduce the number of indices needed to check for graphical testing in many cases [15].

A major difference in the graphs we produce versus the graphs of the Li et al. algorithm is that we do not bother to ensure that the graphs are connected. Thus, we are actually giving an upper bound on s_{max}. We can compute a lower bound on s_{max} by by constructing a connected graph from the graph produced by our approximation algorithm by using a common component merging algorithm [12]. By iteratively performing switches with an edge in a cycle in one component with an arbitrary edge in another component, we can consolidate all the components into one connected graph. This connected graph will have an s value that is a lower bound on s_{max}.

In order to test the quality of our approximation scheme, we generated a set of 158 degree sequences ranging from 50 nodes to 4000 nodes. Each sequence is a graphical sequence generated from a power law distribution with $\alpha = 2.1$. Using these sequences, we generated s_{max} values using the Li et al. algorithm and then a set of approximations using our algorithm. The results of this comparison are shown in figures 2 and 3. In figure 2, we see both the calculated s_{max} values for the degree sequences and the difference between the approximation and the true value. In figure 3, we see the relative error of our approximation. This test shows that, at least on relatively small degree sequences, this approximation gives good and fast results.

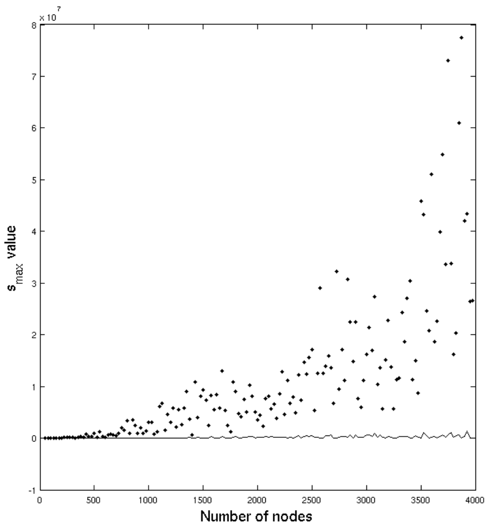

Figure 2: Plot of the s_{max} values for each of the degree sequences and the difference between approximation and true value

5 Conclusions

This paper provides a practical scheme for generating graphs with a given S value. The basic ideas presented here for generating networks constrained by the S metric can be extended to almost any network metric. The authors have had success in implementing the schemes mentioned in this paper and using them in order to study the properties of different networks. We are currently considering applications of these modeling techniques in simulations of the Border Gateway Protocol (BGP) routing system [16].

Future work in the generation of random networks involves two theoretical open problems resulting from the work in this paper. The first is to investigate when a Markov chain for a given ϵ, using the construction in section 3, is irreducible and

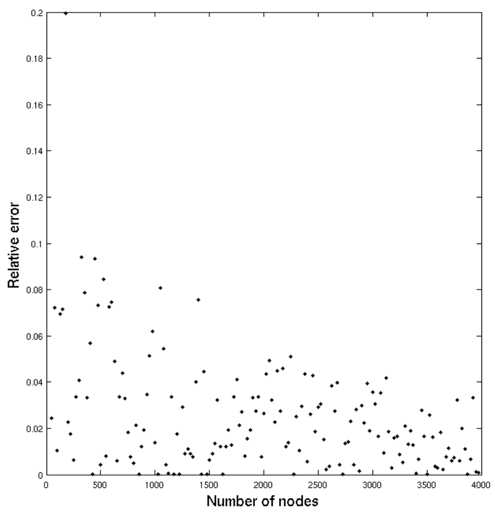

Figure 3: Relative error for each of the data points between the true s_{max} value and the approximation

thus is ergodic. A result in this area could produce an algorithm to uniformly select a random graph G with a given degree sequence ω and an s value of $[s_p - \epsilon, s_p + \epsilon]$. This would be valuable for comparing structural properties of real networks versus uniformly selected ones having various constraints.

The second open problem is to establish whether or not the approximation scheme for s_{max} converges and if it does converge, then establishing the rate of convergence. This will give us a confidence in using this approximation scheme for much larger networks where calculating exact s_{max} values is intractable.

Acknowledgment

The authors would like to thank Francis Sullivan and Ron Boisvert for their helpful comments about this paper. This work is a contribution of NIST, an agency of the US government, and is not subject to US copyright.

References

[1] A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” *Science*, vol. 286, pp. 509–512, 1999.

[2] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” *Nature*, vol. 393, pp. 440–442, 1998.

[3] H. Jeong, B. Tombor, R. Albert, Z. Oltvai, and A. L. Barabási, “The large-scale organization of metabolic networks,” *Nature*, vol. 407, no. 6804, pp. 651–654, October 2000.

[4] R. J. Williams and N. D. Martinez, “Simple rules yield complex food webs,” *Nature*, vol. 404, no. 6774, pp. 180–183, March 2000.

[5] L. A. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley, “Classes of small-world networks,” *Proc Natl Acad Sci USA*, vol. 97, no. 21, pp. 11 149–11 152, October 2000.

[6] M. Faloutsos, P. Faloutsos, and C. Faloutsos, “On power-law relationships of the internet topology,” in *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication (SIGCOMM).* New York, NY, USA: ACM, 1999, pp. 251–262.

[7] A. Potanin, J. Noble, M. Frean, and R. Biddle, “Scale-free geometry in OO programs,” *Communications of the ACM*, vol. 48, no. 5, pp. 99–103, 2005.

[8] L. Li, D. Alderson, J. C. Doyle, and W. Willinger, “Towards a theory of scale-free graphs: Definition, properties, and implications,” *Internet Mathematics*, vol. 2, no. 4, pp. 431–523, 2005.

[9] I. Beichl and B. Cloteaux, “Measuring the effectiveness of the s-metric to produce better network models,” 2008, submitted.

[10] G. Dueck and T. Scheuer, “Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing,” *Journal of Computational Physics*, vol. 90, no. 1, pp. 161–175, 1990.

[11] F. Viger and M. Latapy, “Efficient and simple generation of random simple connected graphs with prescribed degree sequence,” in *Proceedings of the Eleventh International Computing and Combinatorics Conference (COCOON)*, ser. Lecture Notes in Computer Science, L. Wang, Ed., vol. 3595. Springer, 2005, pp. 440–449.

[12] C. Gkantsidis, M. Mihail, and E. Zegura, "The Markov chain simulation method for generating connected power law random graphs," in *Proceedings of the Fifth Workshop on Algorithm Engineering and Experiments (ALENEX)*, R. E. Ladner, Ed. SIAM, 2003, pp. 16–25.

[13] D. Randall, "Rapidly mixing Markov chains with applications in computer science and physics," *Computing in Science and Engg.*, vol. 8, no. 2, p. 30, 2006.

[14] J. Blitzstein and P. Diaconis, "A sequential importance sampling algorithm for generating random graphs with prescribed degrees," 2005, submitted to *Annals of Applied Probability.*

[15] A. Tripathi and S. Vijay, "A note on a theorem of Erdős & Gallai," *Discrete Mathematics*, vol. 265, no. 1-3, pp. 417–420, 2003.

[16] K. Sriram, D. Montgomery, O. Borchert, O. Kim, and D. R. Kuhn, "Study of BGP peering session attacks and their impacts on routing performance," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 10, pp. 1901–1915, October 2006.