Chapter 11

Visualizing Large Kronecker Graphs

Huy Nguyen[*], *Jeremy Kepner*[†], *and Alan Edelman*[‡]

Abstract

Kronecker graphs have been shown to be one of the most promising models for real-world networks. Visualization of Kronecker graphs is an important challenge. This chapter describes an interactive framework to assist scientists and engineers in generating, analyzing, and visualizing Kronecker graphs with as little effort as possible.

11.1 Introduction

Kronecker graphs are of interest to the graph mining community because they possess many important patterns of realistic networks and are useful for theoretical analysis and proof (see [Leskovec et al. 2010] and Chapters 9 and 10). Once the model is fitted to the real networks, many applications can be built on top of Kronecker graphs, including graph compression, extrapolation, sampling, and anonymization. Moreover, there are efficient algorithms that can find Kronecker graphs that match important patterns of real networks [Leskovec et al. 2010]. Nevertheless, our understanding of Kronecker graphs is still limited. Chapter 9 (see also [Kepner 2008]) has shown that a simple combination of bipartite plus identity

[*]MIT CSAIL, 32 Vassar Street, Cambridge, MA 02139 (huy2n@mit.edu).

[†]MIT Lincoln Laboratory, 244 Wood Street, Lexington, MA 02420 (kepner@ll.mit.edu).

‡MIT Math Department, 77 Massachusetts Ave., Cambridge, MA 02139 (edelman@mit.edu).

This work is sponsored by the Department of the Air Force under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

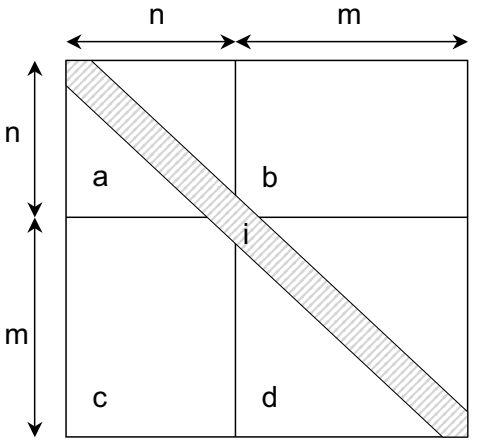

Figure 11.1. The seed matrix G.

graphs can generate a rich class of graphs that have many important patterns of a realistic network.

We have developed an interactive toolkit that assists scientists and algorithm designers in generating, analyzing, and visualizing Kronecker graphs. On a commodity workstation, the framework can interactively generate and analyze Kronecker graphs with millions of vertices. In order to visualize Kronecker graphs, we implement a visualizing algorithm on a parallel system that can then efficiently drive a large 250 megapixel display wall [Hill 2009]. This system allows for effective visualization of graphs of up to 100,000 vertices. Moreover, the framework is designed in an intuitive and interactive fashion that is easy to use and does not require much experience from the users in working with Kronecker graphs. We believe this tool can be a potential first step for anyone who wants to use Kronecker graphs as a model for their own networks.

11.2 Kronecker graph model

The working model of Kronecker graphs in our framework is a generalization of the bipartite stochastic model (see Chapter 9). In particular, let $\mathbf{G}$ be the seed matrix that is used to generate Kronecker graphs. $\mathbf{G}$ is a linear combination of a bipartite matrix $\mathbf{B}(n, m)$ and a diagonal matrix $\mathbf{I}$ (see Figure 11.1). $\mathbf{B}(n, m)$ is a four-quadrant matrix with size $(m+n) \times (m+n)$. The values of the entries in each of the quadrants are a, b, c, and d. In the diagonal matrix, all entries in the main diagonal have value i.

This simple model covers a range of Kronecker graphs. The stochastic model presented in [Leskovec et al. 2010] is a special case with $m = n = 1$ and $i = 0$. The model in [Kepner 2008] (where $\mathbf{G}$ is the union of a bipartite graph and an identity graph) is also a special case of our model with $a = d = 0$ and $b = c = i = 1$.

11.3 Kronecker graph generator

Let N and M be the desired number of vertices and edges of the Kronecker graph. If $N = (m+n)^k$, it is simple to generate a graph with the desired number of vertices. Specifically, we consider the $N \times N$ matrix $\mathbf{A} = \mathbf{G}^{\otimes k}$ where $\mathbf{G}$ is the seed matrix given above. For any pair of vertices i and j, a directed edge from i to j is created with probability $\mathbf{A}(i, j)$. In case the desired graph is sparse, $M = O(N)$, computing the whole matrix $\mathbf{A}$ is redundant. Instead, the algorithm can just generate the edge list directly from the seed matrix, with total running time $O(M \log N)$.

However, in case N is not a power of $m + n$, it is unclear how to generate Kronecker graphs from the algorithm above. A simple interpolation algorithm that creates a larger Kronecker graph (with $(m + n)^k$ vertices such that $(m + n)^{k-1} < N < (m + n)^k$) and then selects an appropriate subgraph does not work. Experiments show that simple interpolation produces large jumps (400%) on the edge/vertex (M/N) ratio in the resulting graph (see Figure 11.2). Sudden jumps in the edge/vertex ratio are not consistent with how real-world graphs grow.

The jumps in the edge/vertex ratio can be reduced by selecting the subgraph more intelligently using our "organic growth" interpolation algorithm. For a given desired N, our algorithm picks $(m+n)^k$—the smallest power of $m+n$ that is larger than N. Then, we generate an edge list (directly from the seed matrix) for the graph of size $(m+n)^k \times (m+n)^k$. Now, instead of taking a subgraph of size N, we randomly shuffle the labels of the vertices and then pick the N vertices with highest degree. As shown in Figure 11.2, generating Kronecker graphs in this way reduces the interpolation error of the edge/vertex ratio significantly.

11.4 Analyzing Kronecker graphs

Analyzing Kronecker graphs is the main feature of our framework. Once a graph is generated, it can be analyzed by three different methods: graph metrics, graph view, and graph organic growth. The graph metrics are a set of statistics about the structure of the graph that can be used to compare the similarity between the generated Kronecker graph and the target real network. The graph view helps users observe the generated graph from different perspectives and thereby identify important properties of the graph. Finally, the graph organic growth simulates the growing (or shrinking) process of a network graph using the previously described interpolation algorithm.

11.4.1 Graph metrics

We compute a set of statistics that can be used to derive many of the important metrics of a given graph.

- *Degree distribution power law exponent*: The degree distribution of a graph is a power law if the number of vertices with degree d in the graph is proportional to $d^{-\lambda}$ where $\lambda > 0$ is the *power law exponent*. Both the real networks and the Kronecker graphs can exhibit power law degree distribution.

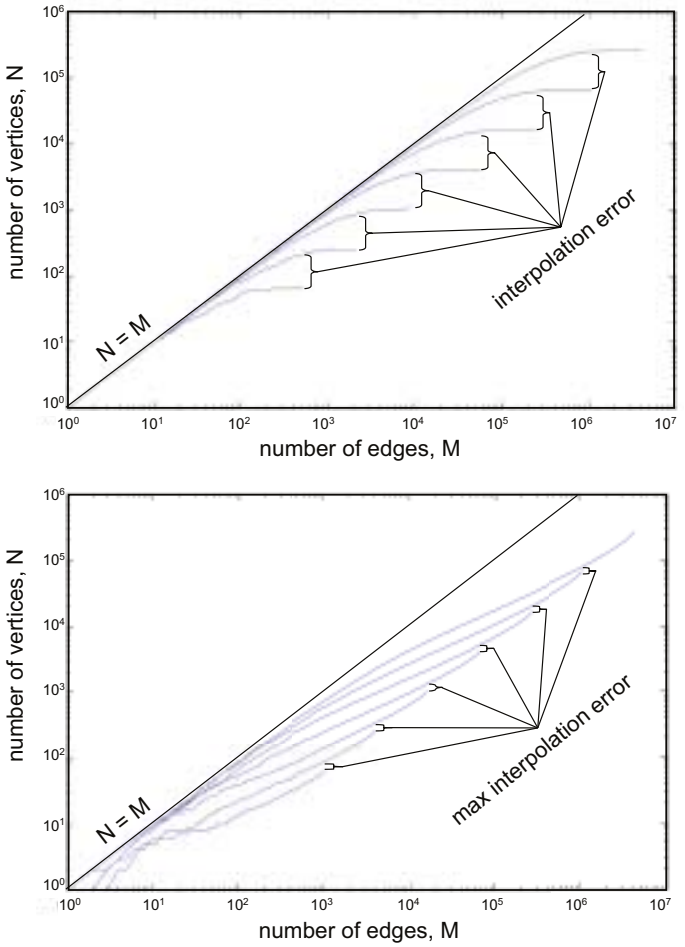

Figure 11.2. Interpolation algorithm comparison.
Desired number of vertices versus the resulting number of edges for the simple interpolation algorithm (top) and our organic growth interpolation algorithm (bottom). The simple interpolation approach can produce a 400% jump in the number of edges when the desired number of vertices N is very close to $(n+m)^k$. The organic growth interpolation algorithm randomizes the vertex labels and selects the highest degree nodes to produce a much smoother curve with smaller jumps at these boundaries.

- *Densification power law exponent*: The number of edges M may be proportional to N^α where α is the densification power law exponent. Similar to the degree distribution power law exponent, this exponent can also be used as a metric of the graph [Leskovec et al. 2005].

- *Effective graph diameter*: The *effective graph diameter* was proposed in [Leskovec et al. 2005] as a robust measurement of the maximum distance between vertices in the graph. It is observed that many real-world networks have relatively small effective graph diameter (see [Leskovec et al. 2005, Albert 2001]). The diameter of the graph is not used here since it is susceptible to outliers.
- *Hop plot*: The function $f : k \to f(k)$, which is the fraction of pairs of vertices that have distance k in the graph [Palmer et al. 2002].
- *Node triangle participation*: The function $g : k \to g(k)$, which is the number of vertices that participate in k triangles in the graph [Tsourakakis 2008].

11.4.2 Graph view

Graph view is a visual way to study the structure of a Kronecker graph. The view is a 2D image of its 0/1 adjacency matrix under some permutation of the vertices. By adding more than one permutation to the framework, we hope that the images they create can give the users different perspectives about the graph structure and can help identify useful patterns. For example, if we view a Kronecker graph in the order it was generated, it is difficult to realize that its degree distribution obeys the power law. However, if we view the graph in degree sorted order, the power law property of its degree distribution can be easily noticed. Our framework allows a user to visualize the generated graph in four different permutations:

- Degree sorted: This view corresponds to vertices in nonincreasing order of degree.
- Randomized: This is the view taken from a random permutation.
- As generated: This is the view where the original order of vertices is used.
- Bipartite: This view exploits the inherent structure of a bipartite generating graph. The permutation takes advantage of the near-bipartiteness of the seed graph to efficiently detect the highly connected components in the graph and separate them from others. For more details on this permutation, see Chapter 10.

11.4.3 Organic growth simulation

In addition to the graph metrics and graph view, which only work with static Kronecker graphs, it is possible to use the organic growth interpolation algorithm to simulate how a graph might grow (or shrink). The simulation can show how a graph has been growing up from a single vertex to the current state and beyond. The main application of this feature is network extrapolation [Leskovec et al. 2010]. Given a real-world network and a Kronecker graph G that models that network, then by applying organic growth simulation on G, we can look into the future to see how the network might evolve. Similarly, organic growth simulation can also help us look into the past to see how the network might have looked in its early stages.

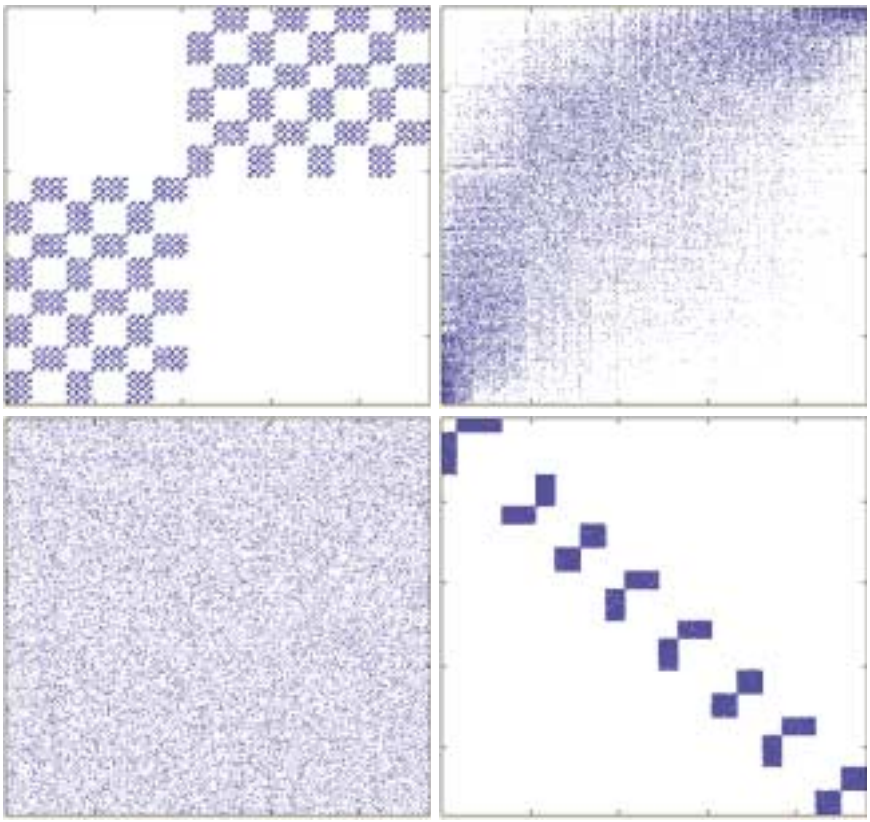

Figure 11.3. Graph permutations.
Views of different permutations of a Kronecker graph generated with a bipartite generating graph. Top left: As generated. Bottom left: Randomized. Top right: Degree sorted. Bottom right: Bipartite. The bipartite permutation clearly shows that the resulting graph consists of eight disconnected bipartite graphs.

11.5 Visualizing Kronecker graphs in 3D

Good visualization is an important part of an analytical framework. In this section, we describe our tool to visualize a Kronecker graph by projecting it onto the surface of a sphere. The idea of embedding a Kronecker graph onto a sphere was proposed and proved effective by Gilbert, Reinhardt, and Shah [Gilbert et al. 2007]. However, their embedding algorithm, which was designed for general graphs, did not take advantage of Kronecker graphs' structure. In our algorithm, we use the bipartite clustering method (as in the bipartite view) to partition the graph into well-connected components (see Figures 11.3 and 11.4) and embed them onto the sphere. As a result, the visualization quality has been improved significantly compared to the Fiedler mapping method (see Figure 11.5).

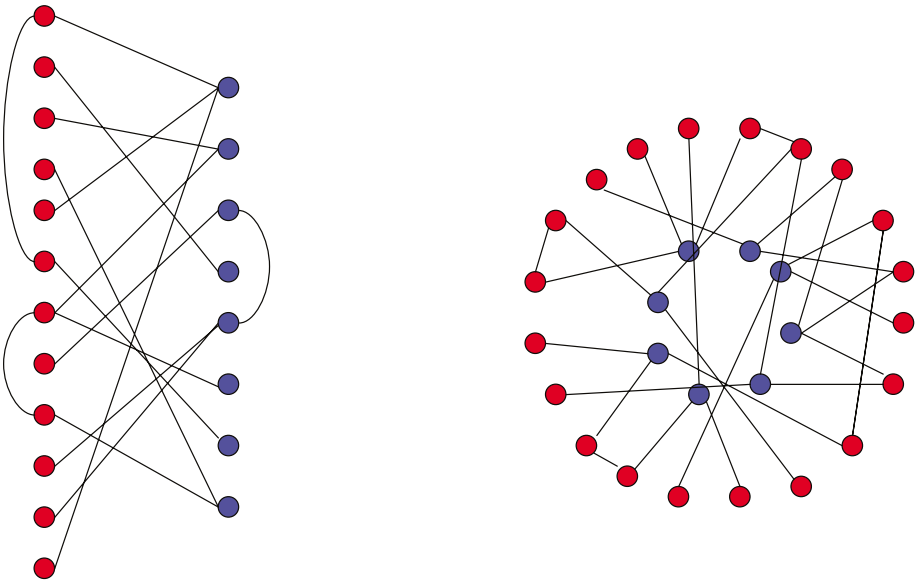

Figure 11.4. Concentric bipartite mapping.
Left: A near bipartite subgraph. Right: Mapping of subgraph on concentric circles.

11.5.1 Embedding Kronecker graphs onto a sphere surface

Given a Kronecker graph, our visualization first partitions such a graph into dense near-bipartite subgraphs using the bipartite permutation (see Chapter 10). Each subgraph is organized into a pair of concentric circles (see Figure 11.4), and these subgraphs are then placed onto a sphere so that they do not overlap. More specifically, for each such subgraph $\mathbf{B}(n, m)$, $\mathbf{B}$ comprises two nearly disjoint sets with n and m, where $n > m$. The subgraph is mapped onto two concentric circles such that n points are in the outer circle and m points are on the inner circle (see Figure 11.5). All the concentric circles are embedded on a sphere surface by using the Golden Section spiral method [Rusin 1998], which guarantees that the circles do not intersect and are evenly distributed. Because the majority of edges in the graph are internal to the subgraph, the visualization is pleasing to the eye and the overall structure of the Kronecker graph can be seen clearly.

11.5.2 Visualizing Kronecker graphs on parallel system

As the framework is designed to work with very large Kronecker graphs (up to 100,000 vertices), it is not practical to visualize them on a commodity workstation. Therefore, we implement our three-dimensional (3D) visualization algorithm on a parallel system with 60 display panels (2560×1600 pixel each) and 30 computational nodes (each node is responsible for 2 display panels), see Figure 11.6.

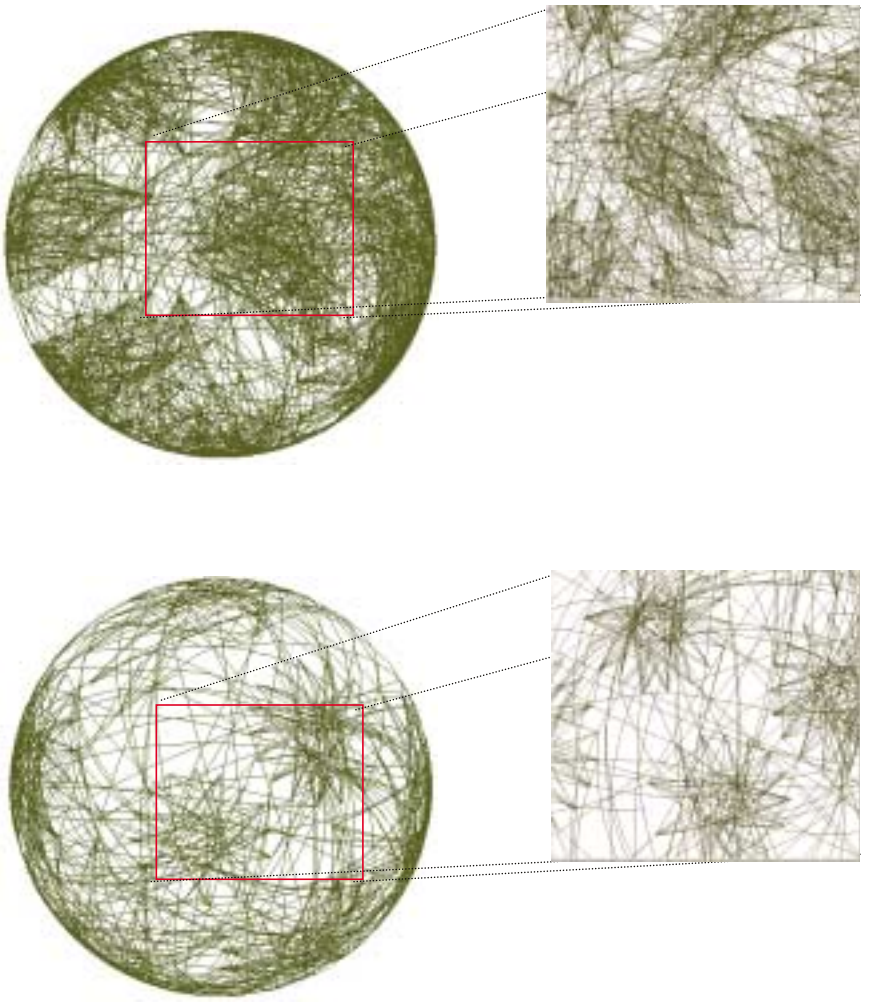

Figure 11.5. Kronecker graph visualizations.
Top: Fiedler mapping of a Kronecker graph onto a sphere. Bottom: Mapping of Kronecker graph with bipartite clustering method.

Figure 11.7 shows how the 3D visualization is designed on the parallel system. First, the graph is distributed to all computational nodes. Then, for each node, all vertices and edges of the graphs that are not visible on that node will be removed. Finally, visible edges and vertices are mapped onto a sphere surface using the algorithm above and rendered on the displaying panels.

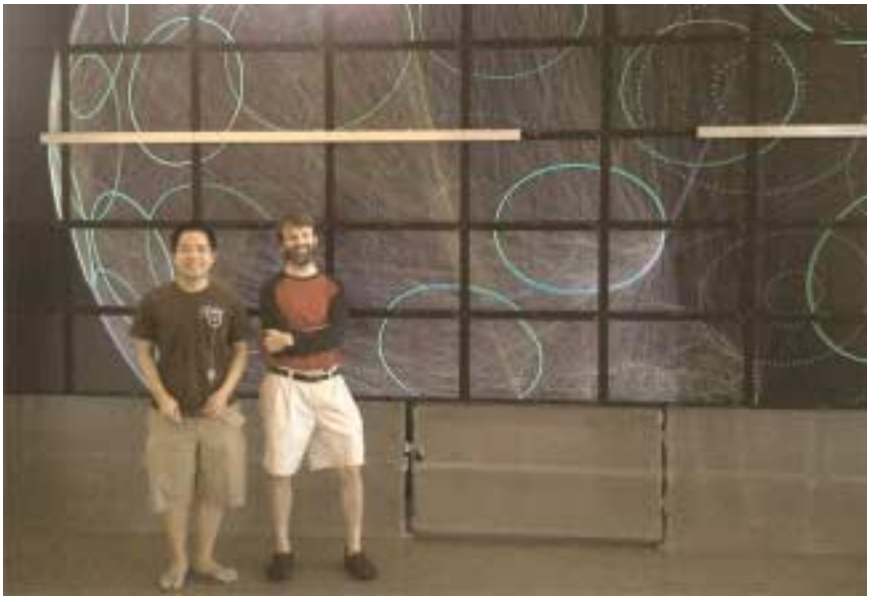

Figure 11.6. Display wall.
Authors with 250 megapixel display wall showing a 100,000-vertex graph.

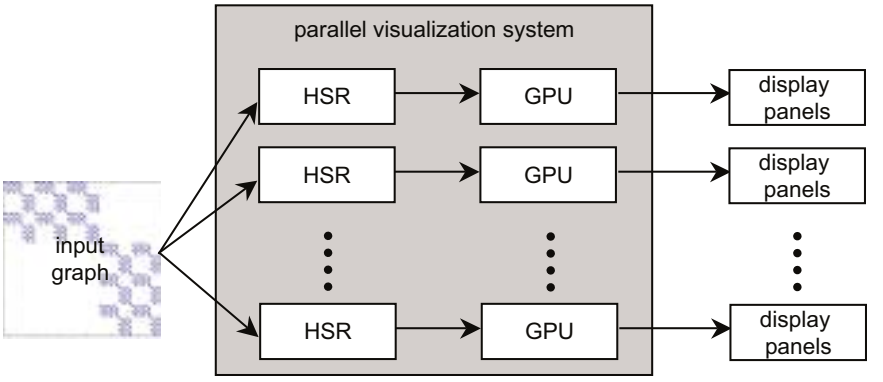

Figure 11.7. Parallel system to visualize a Kronecker graph in 3D.
Components consist of input graph, hidden surface removal (HSR), graphics processing units (GPU), and video display panels.

References

[Albert 2001] R.Z. Albert and A.-L. Barabsi. Statistical mechanics of complex networks. *Reviews of Modern Physics,* 74:47–97, 2002.

[Gilbert et al. 2007] J.R. Gilbert, S. Reinhardt, and V. Shah. An interactive environment to manipulate large graphs. In *Proceedings of the 2007 IEEE International Conference on Acoustics, Speech, and Signal Processing,* 4:IV-1201–IV-1204 2007.

[Hill 2009] C. Hill. The Darwin Project. http://darwinproject.mit.edu/

[Kepner 2008] J. Kepner. Analytic theory of power law graphs. In *SIAM Parallel Processing 2008*, Minisymposium on HPC on Large Graphs, Atlanta, GA, 2008.

[Leskovec et al. 2010] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research*, 11:985–1042, 2010.

[Leskovec et al. 2005] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD '05)*, 177–187, 2005.

[Palmer et al. 2002] C.R. Palmer, P.B. Gibbons, and C. Faloutsos. ANF: A fast and scalable tool for data mining in massive graphs, In *Proceedings of the International Conference on Knowledge Discovery in Data Mining (KDD '02)*, 81–90, 2002.

[Rusin 1998] D. Rusin. Topics on sphere distributions. 1998. http://www.math.niu.edu/~rusin/known-math/95/sphere.faq.

[Tsourakakis 2008] C.E. Tsourakakis. Fast counting of triangles in large real networks without counting: Algorithms and laws. In *Proceedings of the IEEE International Conference on Data Mining*, 608–617, 2008.