

Drawing power law graphs

Reid Andersen * Fan Chung * † Linyuan Lu *

Abstract

We present methods for drawing graphs that arise in various information networks. It has been noted that many realistic graphs have power law degree distribution and exhibit the small world phenomenon. Our algorithm is influenced by recent developments on modeling and analysis of such power law graphs.

1 Introduction

Although graph theory has a history of more than 250 years, it was only recently observed that many realistic graphs from disparate areas satisfy the so-called “power law”. Graphs with power law degree distribution are prevalent in the Internet, in communication networks, social networks and in biological networks [1, 2, 4, 5, 6, 7, 8, 11, 13, 16, 18, 19, 21, 25]. For a fixed value $\beta > 1$, we say that a graph is a power law graph with exponent β if the number of vertices of degree k is proportional to $k^{-\beta}$.

In addition to the power law degree distribution, many realistic graphs exhibit the so-called “small world phenomenon” which refers to two distinct properties — small average distance and the clustering effect. An experiment by Stanley Milgram [20] titled “The small world problem” indicated that any two strangers have small distance, in that they are likely to be connected by a short chain of intermediate acquaintances. The clustering effect implies that any two nodes sharing a neighbor are likely to be adjacent.

It was shown in [9] that a random power law graph has small average distance and small diameter. (Here, average distance is the average of the distances between pairs of nodes that are connected, and the diameter is the maximum distance between connected pairs.) In particular, a random power law graph with exponent β , where $2 < \beta < 3$, contains a dense subgraph, called the “core”, with $n^{c/\log \log n}$ vertices. Almost all vertices are within distance $\log \log n$ of the core although there are vertices at distance $\log n$ from the core [9].

The clustering effect in realistic networks is usually determined by local connectivity and is not captured by standard random graph models. Previous approaches to modeling the clustering effect involve adding random edges to a grid or Euclidean plane [23, 24, 17, 12]. The global graph is modeled by a random power law graph and the local graph has the property that the endpoints of every edge are joined by at least l edge-disjoint paths each of length at most k , for some fixed parameters k and l . It was shown that these hybrid graphs have average distance and diameter of order $O(\log n)$ where n is the number of vertices. In [3], a local graph defined by network flow was introduced, along with an efficient algorithm for extracting the local graph from a given graph. The extraction of the local graph was shown to be robust, in that for a graph generated by the hybrid model, the original local graph is recovered with a small error bound.

In this paper, we present a graph drawing algorithm that takes advantage of our understanding in modeling and analyzing realistic graphs. Our algorithm can be used for drawing general graphs but it is particularly suitable for drawing power law graphs that exhibit the small world phenomenon. We utilize results for the hybrid graph model which are relevant to producing good drawing. A key method is to partition a given graph into a nested sequence of local graphs which reflect these local communities. We also use various structural properties of a random power law graph, such as, the shape of an “octopus”, having

*University of California San Diego, randers@math.ucsd.edu

†Research supported in part by NSF Grants DMS 0100472 and ITR 0205061

a dense core. A dense random graph is usually not amenable to most drawing methods. By separating the core and adjusting the weights of the edges, we can then focus on local graphs and sparse tree-like subgraphs of the global graph, which can be efficiently dealt with.

The paper is organized as follows. In the next section, we give definitions for weighted graphs, quotient graphs, local flow and local graphs. In Section 3, we discuss algorithms for extracting local graphs and identifying communities. The drawing algorithms will be described in Section 4 and examples will be illustrated in Section 5.

2 Preliminaries

2.1 Weighted graphs and Quotient Graphs

Although our input graphs are unweighted, our algorithm will form weighted graphs by collapsing connected components into single vertices. A weighted graph is a simple graph G together with a vertex weight function $w_G(v)$ and an edge weight function $\phi_G(e)$. Suppose that $V(G)$ has a partition $V(G) = C_1 \cup C_2 \cup \dots \cup C_k$. The quotient graph Q is defined as follows. The vertices of Q are communities C_1, \dots, C_k , and we set

$$w_Q(C_k) = \sum_{u \in C_k} w_G(u).$$

$$\phi_Q(C_i, C_j) = \sum_{u \in C_i, v \in C_j} \phi_G(u, v).$$

There is an edge between C_i and C_j if $\phi_Q(C_i, C_j) > 0$.

2.2 Local Flow and Local Graphs

Given a weighted graph with edge capacity function ϕ , we will define a notion of local connectivity between vertices. We will say a path is *short* if it has length less than or equal to ℓ . A short flow is a positive linear combination of short paths where no edge carries more than its capacity. The maximum short flow problem can be viewed as a linear program, and can be computed in polynomial time using nontrivial but relatively efficient algorithms for fractional packing (See 2.3).

Definition 1 (Short Flow) *A short flow is a feasible solution to the following linear program. The flow connectivity $f(u, v)$ between two vertices is the maximum value of any short flow, which is the optimum value of the following LP problem. Let P_ℓ be the collection of short u - v paths, and let P_e be the collection of short u - v paths which intersect the edge e .*

$$\begin{aligned} & \text{maximize} && \sum_{p \in P_\ell} f_p && (1) \\ & \text{subject to} && \sum_{p \in P_e} f_p \leq \phi(e) && \text{for each } e \in L \\ & && f_p \geq 0 && \text{for each } p \in P_\ell \end{aligned}$$

We say two vertices u and v are (f, ℓ) -connected if there exists a short flow between them of size at least f . We say a graph L is an (f, ℓ) -local graph if for each edge $e = (u, v)$ in L , the vertices u and v are (f, ℓ) -connected in L .

2.3 Computing the maximum short flow

The problem of finding the maximum short flow between u and v in a graph G with given edge capacities $\phi(e)$ can be viewed as a fractional packing problem, as introduced by Plotkin, Shmoys, and Tardos [22]. A

fractional packing problem has the form

$$\max\{\mathbf{c}^T \mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \succeq \vec{\mathbf{0}}\}.$$

To view the maximum short flow as a fractional packing problem, first let $G(u, v)$ be a subgraph containing all short paths from u to v . For example, we may take $G(u, v) = N_{\ell/2}(u) \cup N_{\ell/2}(v)$. Let A be the incidence matrix where each row represents an edge in $G(u, v)$ and each column represents a short path from u to v . Let $\mathbf{b} = \phi$, and $\mathbf{c} = \vec{\mathbf{1}}$.

Using the algorithm of Garg and Könemann in [14] for general fractional packing problems, one can obtain a $(1 - \epsilon)^{-2}$ -approximation to the maximum short flow in time $O(M^2 \ell \lceil \frac{1}{\epsilon} \log_{1+\epsilon} M \rceil)$, where M is the number of edges in $G(u, v)$.

3 Extracting the Local Graph

For a given graph, we wish to extract the largest (f, ℓ) -local subgraph. We define $L_{f, \ell}(G)$ to be the union of all (f, ℓ) -local subgraphs in H . By definition, the union of two (f, ℓ) -local graphs is an (f, ℓ) -local graph, and so $L_{f, \ell}(G)$ is in fact the unique largest (f, ℓ) -local subgraph in G . We remark that $L_{f, \ell}(G)$ is not necessarily connected. The following simple greedy algorithm computes $L_{f, \ell}(G)$ in any graph G using $O(m^2)$ max-short-flow computations.

Extract:

Input: G, f, ℓ

If there is an edge $e = (u, v) \in G$ where u, v are not (f, ℓ) -connected in G , remove e from G

When no further edges can be removed, output G .

The number of max-short-flow computations can be reduced by using a standard random sampling approach if we are willing to accept approximate local graphs. We say L is an α -approximate (f, ℓ) -local graph if $L(f, \ell) \subseteq L$, and at most an α -fraction of the edges in L are not (f, ℓ) -connected. The following algorithm extracts a series of approximate local graphs.

Approximate Extract:

Input: $G, \ell, \{f_1 \leq \dots \leq f_k\}$

Let m be the number of edges in G .

For $i = 1 \dots k$:

 Repeat until no edge is removed for $\frac{1}{\alpha} \log \frac{mk}{\delta}$ consecutive attempts:

 Pick an edge $e = (u, v)$ from G uniformly at random.

 If u, v are not (f_k, ℓ) -connected, remove (u, v) from G .

 Then Output $L_k = H$, reset m to be the number of edges in L_k , and continue.

Stop when graphs $L_1 \supseteq \dots \supseteq L_k$ have been output

Since at most m edges are removed from G and there are at most $\frac{1}{\alpha} \log \frac{mk}{\delta}$ attempted removals for every edge removed, **Approximate Extract** performs at most $\frac{m}{\alpha} \log \frac{mk}{\delta}$ max-short-flow computations.

Theorem 1 (Approximate Local Graphs) *Given G, ℓ , and $\{f_0 \leq \dots \leq f_k\}$, let $L_1 \supseteq \dots \supseteq L_k$ be the output of **Approximate Extract**. With probability at least $1 - \delta$, each of the graphs L_i is an α -approximate (f_i, ℓ) -local graph.*

Proof: Given $i \in [1, k]$, let $e_1 \dots e_J$ be the edges removed from L_{i-1} to obtain L_i . Let m_i be the number of edges in L_{i-1} and note that $J \leq m_i$. Let T_j be the number of attempts between the removal of the e_{j-1} and e_j . If L_i is not an α -approximate local graph, then some T_j must be at least $\frac{1}{\alpha} \log \frac{m_i k}{\delta}$ when at least an

α -fraction of the edges remaining in G were not (f_i, ℓ) -connected. For a given j , this occurs with probability at most

$$(1 - \alpha)^{T_j} \leq e^{-\alpha T_j} \leq e^{-\log \frac{m_i k}{\delta}} \leq \delta m_i^{-1}/k.$$

Since $J \leq m_i$, the probability that this occurs for any T_j is at most δ/k . The probability that a bad T_j occurs for any L_i is at most δ , and the result follows.

4 A general algorithm for drawing power law graphs

In this section we describe a framework for producing drawings of power law graphs that reflect local connectivity. Our algorithm uses as a subroutine a force-based drawing method which we will describe in section 4.2, but other graph drawing algorithms can be used in its place. Various pre and post-processing techniques may also be used to improve the results. Although these drawing algorithms are motivated by the structure of power law graphs, they can be applied to general graphs as well.

4.1 Drawing multi-level local structure

Given an input graph G , there is a simple way to highlight a single level of local structure. Compute the local graph $L_{f,\ell}$ using the **Extract** or **Approximate Extract** algorithm for some choice of f, ℓ . The edges in $G \setminus L_{f,\ell}$ are considered global edges. Use force-based methods to produce a drawing of $L_{f,\ell}$, and then add in the global edges drawn with a lighter color.

In the algorithm **MULTILEVEL-DRAW**, this main idea is extended to reflect multiple levels of local structure. Special preprocessing steps take into account the dense core of the power law graph and its tree-like structures. The algorithm consists of three stages.

MULTILEVEL-DRAW:

- i) Obtain a nested sequence of local communities
- ii) Assign edge-weights based on membership in communities
- iii) Recursively draw local communities and tree-like quotient graphs

(i) Obtain a nested sequence of local communities

For a given graph G , choose a fixed ℓ and use the **Approximate Extract** algorithm to compute $L_{0,\ell} \supseteq L_{1,\ell} \supseteq \dots \supseteq L_{K,\ell}$ until the graph $L_{K,\ell}$ contains only isolated vertices. Let $\Pi_0 \supseteq \dots \supseteq \Pi_K$ be the partitions induced by the connected components in $L_{0,\ell}, \dots, L_{K,\ell}$, which divide G into subgraphs that are successively smaller and more locally connected. It will be helpful to consider the tree T with levels $T_1 \dots T_K$ where each connected component in Π_k is in T_k , and its ancestor is the component in Π_{k-1} which contains it. For each component C in T_k , we define the quotient graph $Q(C)$ by collapsing each of its subcomponents in T_{k+1} to a single vertex.

The above method can be modified by choosing ℓ in a flexible way. Instead of a fixed ℓ , we choose a constant c , where $0 < c < 1$. Let ℓ_i denote the least integer so that L_{i,ℓ_i} contains at least a fraction c of the number of edges in $L_{i-1,\ell_{i-1}}$.

(ii) Assign edge-weights based on membership in communities

We can use the local graphs to assign weights to edges as follows. Let the edge set E be partitioned into:

$$E = E_0 \cup E_1 \cup \dots \cup E_K,$$

where $e \in E_k$ if e is inside some cluster in $T_0 \dots T_k$, but between clusters in $T_{k+1} \dots T_K$. Pick a strength function which assigns strength $s(k)$ to every edge in E_k . The strength function should be increasing to give greater strength to more highly connected edges, for example $s(k) = c^k$ for some $c > 0$.

This information may also be useful in other settings. We can give meaningful weights to an unweighted graph, and then apply standard drawing algorithms. Also, given any drawing we can draw the weaker edges in lighter colors to emphasize the local structure of the graph. This is especially useful when drawing power law graphs, where we can not expect to produce a good drawing of the dense core of random-like global edges.

(iii) *Recursively draw local communities and tree-like quotient graphs.*

To produce a drawing of a connected component C in T_k , we can use a typical recursive algorithm: We produce a drawing of $Q(C)$, and drawings of each of the subcomponents of C in T_{k+1} . The drawings of the subcomponents are obtained by recursion, and the drawings of the quotient graphs are obtained by our force-based algorithm. To combine into a single drawing, we replace each vertex in the drawing of $Q(C)$ with the drawing of the corresponding subcomponent, scaled appropriately, and apply our force-based algorithm to the resulting drawing as a post-processing step.

The above recursive algorithm works best when the quotient graphs are not too dense. In practice, for random-like power law graphs, the quotient graphs that arise can have a dense core, but most vertices and edges have small weight. We make use of this fact to obtain a favorable drawing of a large quotient graph. We first form a maximum-weight spanning tree of the quotient graph by greedily including the heaviest remaining edge which does not create a cycle. We then use our force-based drawing method, or alternatively use standard methods for drawing trees.

4.2 A force-based drawing method

Our algorithms use a standard force-based drawing method, modified for use on graphs with vertex weights $w(v)$ and edge weights $\phi(e)$. We define a repulsive force between every pair of vertices, where the force acting on vertex u due to vertex v is

$$R_{u,v} = \frac{1}{n^2} \frac{u-v}{\|u-v\|^2} w(u)w(v)$$

Each edge also acts as a spring, with the force on a vertex u from the edge $e = (u, v)$ defined to be

$$S_{u,v} = \frac{1}{n} (v-u)\phi(e)$$

The standard force-based approach is to compute the sum of the forces

$$R(u) = \sum_v R(u, v) \quad \text{and} \quad S(u) = \sum_{v \sim u} S(u, v)$$

acting on each vertex and move in the resulting direction at each time step. We have used the following linear approximation heuristic to improve the convergence. We wish to find new coordinates u' for each vertex u such that the repulsive and attractive forces are balanced, $R(u') = -S(u')$. If we let $u' = u + \Delta u$, then, we wish to find Δu such that $R(u + \Delta u) = -S(u + \Delta u)$. We let R_x and R_y be the partial derivatives with respect to the x and y coordinates of u , and instead solve the linear system

$$R(u) + R_x(u)\Delta u_x + R_y(u)\Delta u_y = S(u) + S_x(u)\Delta u_x + S_y(u)\Delta u_y.$$

We then move u by adding the vector Δu thus obtained.

5 Implementation and examples

We have implemented the MULTILEVEL-DRAW algorithm and experimented on several examples. Figure 1 is a resulted drawing for a sparse random graph, generated by the Erdős-Rényi model $G(n, p)$ with $n = 500$ and $p = .004$. Jerry Grossman[15] has graciously provided the data of collaboration graphs, with about 337,000 vertices, and 226,000 edges. We apply the MULTILEVEL DRAW algorithm to the collaboration

graph of the second kind. Namely, each vertex represents an author and each edge represents a joint paper with exactly two authors. Figure 2 is a drawing of the induced subgraph on 2057 authors with Erdős number at most 2. This drawing is a combination of a global graph (as shown in Figure 3) and local graphs (one of which is included in Figure 4).

References

- [1] L. A. Adamic and B. A. Huberman, Growth dynamics of the World Wide Web, *Nature*, **401**, September 9, 1999, pp. 131.
- [2] W. Aiello, F. Chung and L. Lu, A random graph model for massive graphs, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, (2000) 171-180.
- [3] R. Andersen, F. Chung and L. Lu, Analyzing the small world phenomenon using a hybrid model with local network flow, preprint.
- [4] R. B. R. Azevedo and A. M. Leroi, A power law for cells, *Proc. Natl. Acad. Sci. USA*, vol. **98**, no. 10, (2001), 5699-5704.
- [5] Albert-László Barabási and Réka Albert, Emergence of scaling in random networks, *Science* **286** (1999) 509-512.
- [6] A. Barabási, R. Albert, and H. Jeong, Scale-free characteristics of random networks: the topology of the world wide web, *Physica A* 272 (1999), 173-187.
- [7] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tompkins, and J. Wiener, "Graph Structure in the Web," *proceedings of the WWW9 Conference*, May, 2000, Amsterdam. Paper version appeared in *Computer Networks* **33**, (1-6), (2000), 309-321.
- [8] K. Calvert, M. Doar, and E. Zegura, Modeling Internet topology. *IEEE Communications Magazine*, **35(6)** (1997) 160-163.
- [9] F. Chung and L. Lu, Average distances in random graphs with given expected degree sequences, *Proceedings of National Academy of Science*, **99** (2002), 15879-15882.
- [10] F. Chung and L. Lu, The small world phenomenon in hybrid power law graphs *Lecture Note in Physics* special volume on "Complex Network", to appear.
- [11] C. Cooper and A. Frieze, On a general model of web graphs, *Random Structures and Algorithms* Vol. **22**, (2003), 311-335.
- [12] A. Fabrikant, E. Koutsoupias and C. H. Papadimitriou, Heuristically optimized trade-offs: a new paradigm for power laws in the Internet, *STOC* 2002.
- [13] M. Faloutsos, P. Faloutsos, and C. Faloutsos, On power-law relationships of the Internet topology, *Proceedings of the ACM SIGCOM Conference*, Cambridge, MA, 1999.
- [14] N. Garg, J. Konemann, Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *Technical Report, Max-Planck-Institut für Informatik, Saarbrücken, Germany* (1997). <http://citeseer.ist.psu.edu/garg97faster.html>
- [15] Jerry Grossman, Patrick Ion, and Rodrigo De Castro, Facts about Erdős Numbers and the Collaboration Graph, <http://www.oakland.edu/grossman/trivia.html>.
- [16] S. Jain and S. Krishna, A model for the emergence of cooperation, interdependence, and structure in evolving networks, *Proc. Natl. Acad. Sci. USA*, vol. **98**, no. 2, (2001), 543-547.
- [17] J. Kleinberg, The small-world phenomenon: An algorithmic perspective, *Proc. 32nd ACM Symposium on Theory of Computing*, 2000.
- [18] J. Kleinberg, S. R. Kumar, P. Raghavan, S. Rajagopalan and A. Tomkins, The web as a graph: Measurements, models and methods, *Proceedings of the International Conference on Combinatorics and Computing*, 1999.
- [19] S. R. Kumar, P. Raghavan, S. Rajagopalan and A. Tomkins, Extracting large-scale knowledge bases from the web, *Proceedings of the 25th VLDB Conference*, Edinburgh, Scotland, 1999.
- [20] S. Milgram, The small world problem, *Psychology Today*, **2** (1967), 60-67.
- [21] M. E. J., Newman, The structure of scientific collaboration networks, *Proc. Natl. Acad. Sci. USA*, vol. **98**, no. 2, (2001), 404-409.

- [22] S. Plotkin, D. B. Shmoys, and E Tardos, Fast approximation algorithms for fractional packing and covering problems, *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*, 1991, pp. 495–504. <http://citeseer.ist.psu.edu/plotkin95fast.html> ?
- [23] D. J. Watts, *Small Worlds — The Dynamics of Networks between Order and Randomness*, Princeton University Press, New Jersey, 1999.
- [24] D. J. Watts and S. H. Strogats, Collective dynamics of ‘small world’ networks, *Nature* **393**, 440-442.
- [25] E. Zegura, K. Calvert, and M. Donahoo, A quantitative comparison of graph-based models for Internet topology. *IEEE/ACM Transactions on Networking*, **5** (6), (1997), 770-783.

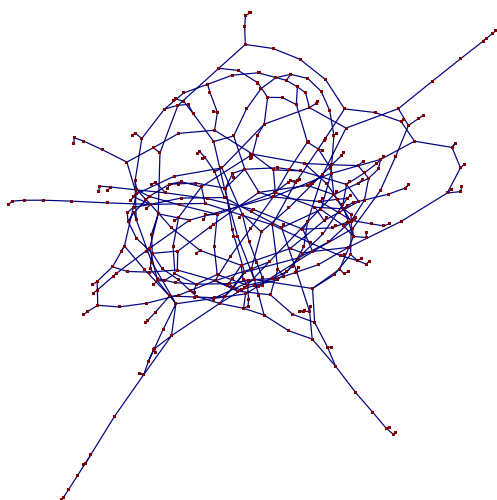


Figure 1: *The giant component of random graph $G(n, p)$ with $n = 500$ and $p = 0.004$.*

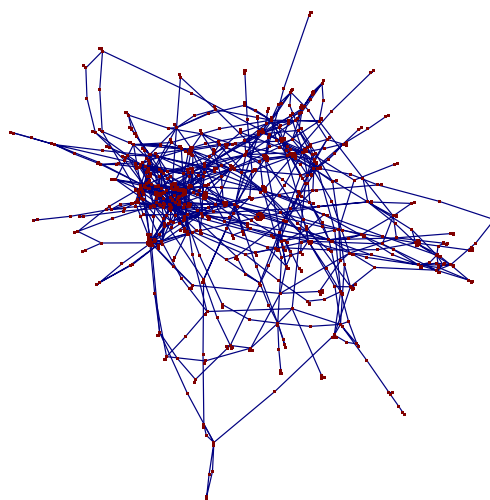


Figure 2: *The induced subgraph of the collaboration graph on authors with Erdős number at most 2.*

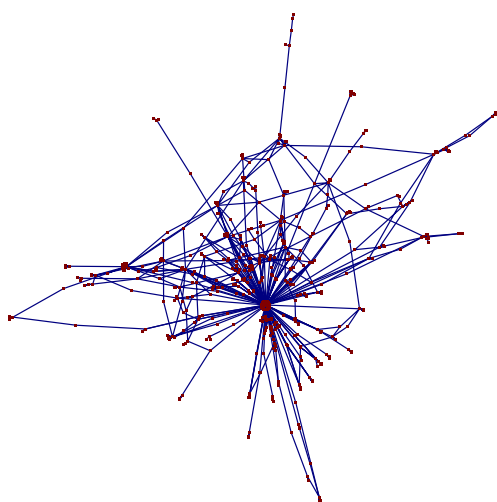


Figure 3: *Global view: the quotient graph of C .*

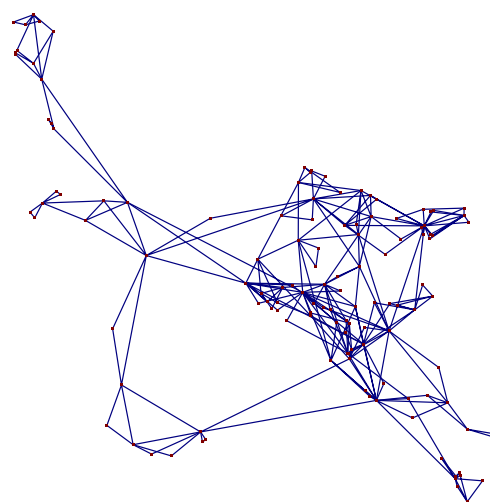


Figure 4: *Local view: the largest local community of C with size 127.*