Weighted Graphs and Disconnected Components

Patterns and a Generator

Mary McGlohon
Carnegie Mellon University
School of Computer Science
5000 Forbes Ave.
Pittsburgh, Penn. USA
mmcgloho@cs.cmu.edu

Leman Akoglu
Carnegie Mellon University
School of Computer Science
5000 Forbes Ave.
Pittsburgh, Penn. USA
lakoglu@cs.cmu.edu

Christos Faloutsos
Carnegie Mellon University
School of Computer Science
5000 Forbes Ave.
Pittsburgh, Penn. USA
christos@cs.cmu.edu

ABSTRACT

The vast majority of earlier work has focused on graphs which are both *connected* (typically by ignoring all but the giant connected component), and *unweighted*. Here we study numerous, real, weighted graphs, and report surprising discoveries on the way in which new nodes join and form links in a social network. The motivating questions were the following: How do connected components in a graph form and change over time? What happens *after* new nodes join a network– how common are repeated edges? We study numerous diverse, real graphs (citation networks, networks in social media, internet traffic, and others); and make the following contributions: (a) we observe that the non-giant connected components seem to stabilize in size, (b) we observe the weights on the edges follow several power laws with surprising exponents, and (c) we propose an intuitive, generative model for graph growth that obeys observed patterns.

Categories and Subject Descriptors: I.6.4 [Computing Methodologies]: Simulation and Modeling—*Model Validation and Analysis*; I.5 [Pattern Recognition]: Miscellaneous

General Terms: Measurement, Theory

1. INTRODUCTION

How do real graphs evolve over time? How do the different components of an entire network form? What happens when we take into account multiple edges and weighted edges? Past work mainly focuses on static snapshots of graphs, where fascinating properties have been discovered, the most striking ones being the 'small-world' phenomenon [30] (also known as 'six degrees of separation' [20]) and the power-law degree distributions [3, 12]. Time-evolving graphs have attracted attention only recently, where even more fascinating properties have been discovered, like *shrinking* diameters, and the so-called *densification power law* [17].

In virtually all the above cases, the analysis focused on the 'giant connected component' (GCC), either explicitly or implicitly, and moreover it ignored multiple links between nodes or weights on edges. Here we will shift our focus to the components that are of moderate size but "disconnected" from the GCC of the undirected graph, which we will refer to as the "next-largest connected components" (NLCCs). We will also look at edge weights, particularly at how weighted edges are added over time.

The questions of interest are:

- *How do the non-giant weakly connected components behave over time?* One person might argue that they grow, as new nodes are being added; and their size would probably remain a fixed fraction of the size of the GCC. Someone else might counter-argue that they shrink, and they eventually get absorbed into the GCC. What is happening, in real graphs?
- *What distributions and patterns do weighted graphs maintain?* How does the distribution of weights change over time– do we also observe a densification of weights as well as single-edges? How does the distribution of weights relate to the degree distribution? Is the addition of weight bursty over time, or is it uniform?
- *Can we produce a generator that will mimic the above behaviors?* The *preferential attachment* model generates a single connected component. Most other generators try to mimic the skewed in- and out-degree distributions, and they also suffer from the same issue. Our goal is to find a generator that mimics skewed degree distribution in unweighted graphs, as well as producing realistic behavior of non-giant connected components.

Answering these questions is important to understand how natural graphs evolve, and to (a) spot anomalous graphs and sub-graphs; (b) answer questions about entities in a network and what-if scenarios; and (c) discard unrealistic graph generators.

Let's elaborate on each of the above applications: Spotting anomalies is vital for determining abuse of social and computer networks, such as link-spamming in a web graph, fraudulent reputation building in e-auction systems [23], detection of dwindling/abnormal social sub-groups in a social-networking site like Yahoo-360 (`360.yahoo.com`), Facebook (`www.facebook.com`) and LinkedIn (`www.linkedin.com`), and network intrusion detection [15]. Analyzing network properties is also useful for identifying authorities and search algorithms [5, 8, 14], for discovering the "network value" of customers for using viral marketing [27], or to improve recommendation systems [4]. What-if scenarios are vital for extrapolation, provisioning and algorithm design: For example, if we expect that the number of links will double within the next year, we should provision for the appropriate hardware to store and process the upcoming queries.

Finally, rules like the upcoming ones in this paper can help us eliminate unrealistic graph generators. Graph generators are also vital, for simulation of algorithms (like computer network routing algorithms), for simulation of rumor (or virus, or influence) propagation, and many other settings. In several such settings, real graphs may be difficult or even impossible to collect: for example a who-believes-whom graph is only in the mind of the human subjects; a who-mails-whom graph may be protected by privacy laws.

Next we present related work (Section 2), background material (Section 3), our observations on un-weighted and weighted graphs, (Sections 5,6) our Butterfly generator model (Section 7), and the conclusions.

2. RELATED WORK

Here we review properties of real-world graphs, as well as several graph generators.

Laws and graph properties: Static, unweighted graphs obey several impressive patterns: they usually have small diameter ('small world' phenomenon), they have skewed degree distributions with power-law tails [2, 12], and have similar power laws with respect to the eigenvalues. A power law is an equation of the form $y = x^a$, with a being the exponent of the power law.

For time-evolving graphs, there is the 'Densification Power Law' [17], where real graphs obey the equation $E(t) = N(t)^a$, where a is the densification exponent. (For graphs studied in this work, a fell between 1.03 and 1.7.)

Additional power laws seem to govern the popularity of posts in citation networks, which drops over time, with power law exponent of -1 for paper citations [26] or -1.5 for blog posts [19]. Park et. al. report both static and dynamic measurements of autonomous systems graphs [24]. Chi et al. studied the evolution of communities over time [9]. The work by Kumar et al. [13] seems to be the only one that studied components other than the giant connected component, and showed that there is significant activity there.

Graph Generators: There are many graph generators, and even a recent survey on them [7]. The oldest and probably the most studied is the Erdos-Renyi model where edges are randomly placed among nodes. Although unrealistic, this model leads to the fascinating phenomenon of *phase transition*: at a critical ratio of edges to nodes, the graph suddenly has high probability to have a 'giant connected component' (GCC). The GCC has size $O(n^{\frac{2}{3}})$ [10].

Additional, more realistic models include the small world model [30], the *preferential attachment model* [1], the *Forest Fire Model* [17] and numerous more (the copying model, the 'winner does not take all' model [25], Heuristically Optimized Trade-offs [11]). We refer to the above as *emergent* generators, because they all have local rules (like preferential attachment), and yet they still manage to produce the macroscopic patterns we observe (small diameter, etc). There is a whole family of non-emergent generators, like degree-sequence matching and the more recent "Kronecker" graphs [16]. However, we focus on emergent graph generators here.

In conclusion, our work and the upcoming discoveries differ from all the earlier work, in the following major aspects: (a) We explicitly focus on the *NLCC*s ('next-largest connected components'), while the overwhelming majority of earlier work ignores them completely. (b) We are the first to discover patterns in *weighted* graphs. (c) We give a natural *emergent* generator, which provably achieves a power law in its out-degree, while still obeying all the other observed laws, old and new.

3. PROPOSED METRICS

A static, unweighted graph G consists of a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$: $G = (\mathcal{V}, \mathcal{E})$. We represent the sizes of $\mathcal{V}$ and $\mathcal{E}$ as N and E. The extensions to weighted, and/or time evolving graphs are straightforward. In such cases, an edge has a weight and/or a timestamp. Let's stay with unweighted graphs, for the moment.

Bipartite graphs, like the movie-actor graph of IMDB, consist of disjoint sets of nodes $\mathcal{V}_1$ and $\mathcal{V}_2$, say, for authors and movies, with no edges among nodes of the same type.

Our goals are to find patterns governing (a) the emergence of a giant connected component, (b) the size of the NLCCs, (c) the behavior of edge and weight additions over time.

For the first, we chose to study the *diameter plot*, that is, the diameter $d(t)$ as a function of time t, because we see it has clear spike when the GCC emerges. (See the first column of Fig. 2 and 3)

We will find that weight-addition over time is bursty and self-similar (i.e. fractal), so we introduce tools, such as the *entropy plot*, to assess the burstiness.

3.1 Diameter

How does the largest connected component of a real graph evolve over time? Do we start with one large CC, that keeps on growing? We propose to use the *diameter-plot* of the graph, that is, its diameter, over time, to answer these questions. For a given (static) graph, its *diameter* is defined as the maximum *distance* between any two nodes, where distance is the minimum number of hops (i.e., edges that must be traversed) on the path from one node to another, ignoring directionality.

Following earlier literature, we estimate the so-called *effective diameter*, which is the 90-percentile of the pairwise distances among all reachable pairs of nodes. Estimating the (effective) diameter is an orthogonal issue. We used sampling to estimate it; alternative methods include ANF [22].

3.2 Burstiness and Entropy Plots

We will show that in weighted graphs, the addition of weights is often bursty. In case that the traffic is *self-similar*, then we can measure the burstiness, using the intrinsic, or *fractal* dimension of the cloud of timestamps of edge-additions (or weight-additions). Let $\Delta W(t)$ be the total weight of edges that were added during the t-th interval, e.g., the total network flow on day t, among all the machines we are observing.

Among the many methods that measure self-similarity (Hurst exponent, etc. [28]), we choose the *entropy plot* [29], which plots the entropy $H(r)$ versus the resolution r. The resolution is the scale, that is, at resolution r, we divide our time interval into 2^r equal sub-intervals, sum the weight-additions $\Delta W(t)$ in each sub-interval k ($k = 1 \ldots 2^r$), normalize into fractions p_k ($= \Delta W(t)/W_{total}$), and compute the Shannon entropy of the sequence p_k: $H(r) = -\sum_k p_k \log_2 p_k$.

If the plot $H(r)$ is linear in some range of resolutions, the corresponding time sequence is said to be *fractal* in that range, and the slope of the plot is defined as the *intrinsic* (or *fractal*) dimension D of the time sequence. Notice that a uniform weight-addition distribution yields D=1; a lower value of D corresponds to a more bursty time sequence like a Cantor dust [28], with a single burst having the lowest D=0: the intrinsic dimension of a point. Also notice that a variation of the 80-20 model, the so called 'b-model' [29], generates such self-similar traffic.

4. DATA DESCRIPTION

We studied several large real networks, described in detail in Table 1.We performed experiments on both uni- and bipartite, and both weighted and unweighted graphs.

Several of our graphs had no obvious weighting scheme: for example, a single paper or patent will cite another only a single time. The graphs that did have weights are also further divided into two schemes, *multi-edges* and *edge-weights*. In the edge-weights scheme, there is an obvious weight on edges, such as amounts in campaign donations, or packet-counts in network traffic. For multi-edges, weights are added if there is more than one interaction between two nodes. For instance, if a blog cites another blog at a given time, its weight is 1. If it cites the blog again later, the weight becomes 2.

The datasets are gathered from publicly available data. *NIPS*[1], *Arxiv* and *Patent* [17] are academic paper or patent citation graphs with no weighting scheme. *IMDB* indicates movie-actor information, where an edge occurs if an actor participates in a movie [3]. *Netflix* is the dataset from the Netflix Prize competition[2], with user-movie links (we ignored the ratings); we also noticed that it only contained users with 100 or more ratings. *BlogNet* and *PostNet* are two representations of the same data, hyperlinks between blog posts [19]. in *PostNet* nodes represent individual posts, while in *BlogNet* each node represents a blog. Essentially, *PostNet* is a paper citation network while *BlogNet* is an author citation network (which contains multi-edges).

NetTraffic records IP-source/IP-destination pairs, along with the number of packets sent, per unit time, and *Oregon* is an autonomous systems network [3]. *Auth-Conf*, *Key-Conf*, and *Auth-Key* are all from DBLP [4], with the obvious meanings. *CampOrg* and *CampIndiv* are bipartite graphs from U.S. Federal Election Commission. They record donations (amounts) between political candidates and organizations, and individuals to organizations [5].

In all the above cases, we assume that edges are never deleted, because edge deletion never explicitly appeared in these datasets.

5. UNWEIGHTED GRAPHS

We tracked several graph properties such as the diameter of the graph, edge additions to the graph and the behavior of the connected components of the graph over time. Here, we report the recurring findings, that hold for all the real world graphs we analyzed and provide additional observations for each specific dataset. Note that in this section only, for the purposes of studying diameter and weakly connected components, we will consider graphs **undirected**, that is whenever there is a link from one node to another, we put an undirected edge between them indicating that an interaction has occured. Our later observations on weighted graphs will return to directed versions of these graphs.

5.1 Diameter-plot and Gelling point

Studying the effective diameter of the graphs, we notice that there is often a point in time when the diameter spikes. Before that point, the graph is more or less in an establishment period, typically consisting of a collection of small, disconnected components. This "*gelling point*" seems to also be the time where the GCC "takes off". After the gelling point, the graph obeys the expected rules, such as the densification power law; its diameter decreases or stabilizes; the giant connected component keeps growing, absorbing the vast majority of the newcomer nodes.

OBSERVATION 1 (GELLING POINT). *Real graphs exhibit a gelling point, at which the diameter spikes and (several) disconnected components gel into a giant component.*

In most of these graphs, both unipartite and bipartite, there are clear gelling points. For example, in *NIPS* the diameter spikes at $t = 8$ years, which is a reasonable time for an academic community to gel. In some networks, we only see one side of the spike, due to data construction (the nature of trace-routes in *Oregon* and *NetTraffic*) or massive network size (*Patent*).

We show full results for *PostNet* in Fig. 1, including the diameter plot (Fig. 1(a)), sizes of the NLCCs (Fig. 1(b)), densification plot (Fig. 1(c)), and the sizes of the three largest connected components in log-linear scale, to observe how the GCC dominates the others (Fig. 1(d)). Results from other networks are similar, and are shown in condensed form for space (Fig. 2 for unipartite graphs, and Fig. 3 for bipartite graphs). The left column shows the diameter plots, and the right column shows the NLCCs, which we describe next.

5.2 Constant/Oscillating NLCCs

We particularly studied the second and the third connected component over time. We notice that, after the gelling point, the sizes of these components *oscillate* over time. Further investigation shows that the oscillation may be explained as follows: new-comer nodes typically link to the GCC; very few of the newcomers link to the 2nd (or 3rd) CC, helping them to grow slowly; in very rare cases, a newcomer links both to an NLCC, as well as the GCC, thus leading to the absorption of the NLCC into the GCC. It is exactly at these times that we have a drop in the size of the 2nd CC: Note that edges are not removed, thus, what is reported as the size of the 2nd CC is actually the size of yesterday's 3rd CC, causing the apparent "oscillation". This intuition forms the basis for our upcoming *Butterfly* model (Sec. 7).

An unexpected (to us, at least) observation is that the largest size these components can get seems to be a constant. This is counter-intuitive – based on random graph theory, we would expect the size of the NLCCs to grow with increasing N. Using scale-free arguments, we would expect the NLCCs to have size that would be a (small, but constant) fraction of

[1]`www.cs.toronto.edu/∼roweis/data.html`
[2]`www.netflixprize.com`
[3]University of Oregon *Route Views* project, `www.routeviews.org`
[4]`dblp.uni-trier.de/xml/`
[5]`www.cs.cmu.edu/∼mmcgloho/fec/data/ fec_data.html`

Name	Uni/bipartite	**Weights**	|N|,|E|,time	**Description**
PostNet	Unipartite	None	250K, 218K, 80 days	Posts from blogs
NIPS	Unipartite	None	2K, 3K, 13 yr.	Citation network from NIPS
Arxiv	Unipartite	None	30K, 60K, 13 yr.	Physics citations
Patent	Unipartite	None	4M, 8M, 17 yr.	Patent citations
IMDB	Bipartite	None	757K, 2M, 114 yr.	Actor-movie network
Netflix	Bipartite	None	125K, 14M, 72 mo.	User-movie ratings
BlogNet	Unipartite	Multi-edges	60K, 125K, 80 days	Social network of blogs based on citations
NetTraffic	Unipartite	Edge-weights (Packet-size)	21K, 2M, 52 mo.	Network traffic
Oregon	Unipartite	None	12K, 38K, 6 mo.	Autonomous systems
Auth-Conf	Bipartite	Multi-edges	17K, 22K, 25 yr.	DBLP Author-to-Conference associations
Key-Conf	Bipartite	Multi-edges	10K, 23K, 25 yr.	DBLP Keyword-to-Conference associations
Auth-Key	Bipartite	Multi-edges	27K, 189K, 25 yr.	DBLP Author-to-Keyword associations
CampOrg	Bipartite	Edge-weights (Amounts)	23K, 877K, 28 yr.	U.S. electoral campaign donations from organizations to candidates (available from FEC)
CampIndiv	Bipartite	Edge-weights (Amounts)	6M, 10M, 22 yr.	Election donations from individuals to organizations

Table 1: The datasets studied in this work.

the size of the GCC – to our surprise, this *never* happened, on any of the real graphs we tried. If some underlying growth does exist, it was small enough to be impossible to observe throughout the (often lengthy) time in the datasets.

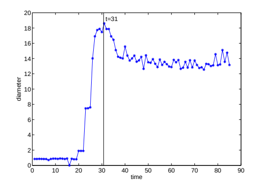
The second columns of Fig. 2 and Fig. 3 show the NLCC sizes versus time. Notice that, after the "gelling" point (marked with a vertical line), they all oscillate about constant value (different for each network). The only extreme cases are datasets with unusually high connectivity. For example, *Netflix*has very small NLCCs. This may be explained by the fact the dataset is masked, omitting users with less than a hundred ratings (possibly to further protect the privacy of the encrypted user-ids). Therefore, the graph has abnormally high connectivity. Also, *Oregon* and *NetTraffic* have unusually high connectivity due to the nature of network traffic– it benefits from having a single GCC, so NLCCs shrink to zero.

Observation 2 (Oscillating NLCCs). *After the gelling point, the secondary and tertiary connected components remain of approximately constant size, with small oscillations.*

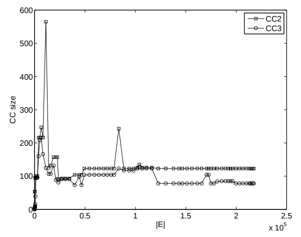
6. WEIGHTED GRAPHS

Here we try to find patterns that weighted graphs obey. In this section we consider graphs to be directed (and impose a single direction in bipartite graphs), as this will be an important consideration on the weights. The typical weighted graph is, say, the *NetTraffic* dataset, which records computer network traffic. The dataset consist of quadruples: (IP-source, IP-destination, timestamp, number-of-packets), where timestamp is in increments of, say, 30 minutes. Thus, we have multi-edges, as well as total weight for each (source, destination) pair. Let $W(t)$ be the total weight up to time t (ie., the grand total of all exchanged packets across all pairs), $E(t)$ the number of distinct edges up to time t, and $E_d(t)$ the number of multi-edges (the d subscript stands for *duplicate* edges), up to time t.

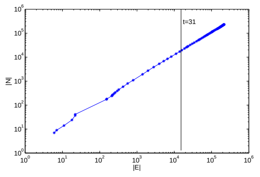
We present three "laws" that our datasets seem to follow: The first is the "weight power law" (WPL) correlating the total weight, the total number of edges and the total number of multi-edges, as they change over time. The second is the "snapshot power law" (SPL), correlating the in-degree with the in-weight, and the out-degree with the out-weight, for all the nodes of a graph, at a given time-stamp. The third



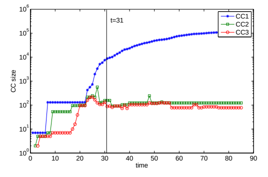
(a) Diameter(t)



(b) CC2 and CC3 sizes



(c) N(t) vs E(t)



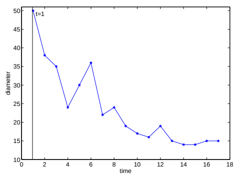
(d) GCC, CC2, and CC3 (log-lin)

Figure 1: Properties of *PostNet* network. Notice that we experience an early gelling point at (a) (diameter versus time), stabilization/oscillation of the NLCC sizes in (b) (size of 2nd and 3rd CC, versus time). The vertical line marks the gelling point. Part (c) gives $N(t)$ vs $E(t)$ in log-log scales - the good linear fit agrees with the Densification Power Law. Part (d): component size (in log), vs time - the GCC is included, and it clearly dominates the rest, after the gelling point.

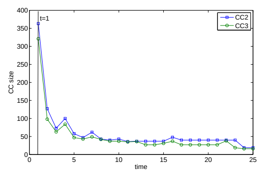
is the "bursty-weight law" (BWL), showing that the weight additions are bursty, over time.

6.1 Weight Power Law (WPL)

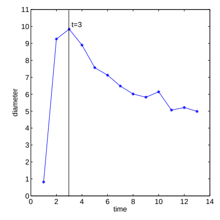
As defined above, suppose we have $E(t)$ total unique edges up to time t (ie., count of pairs that know each other) and $W(t)$ being the total count of packets up to time t. Is there a relationship between $W(t)$ and $E(t)$? If every pair generated k packets, the relationships would be linear: if the count of pairs double, the packet count would double, too. This is reasonable, but it doesn't happen! In reality, the packet count over-doubles, following the "WPL" below. We shall refer to this phenomenon as the "*fortification effect*": more edges in the graph imply super-linearly higher total weight.



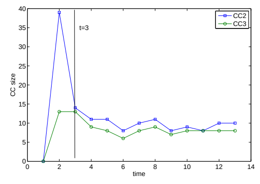
(a) *Patent* Diam(t)



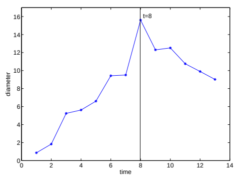
(b) *Patent* NLCCs



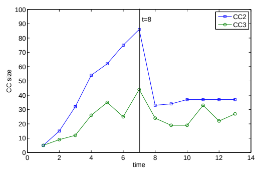
(a) *Arxiv* Diam(t)



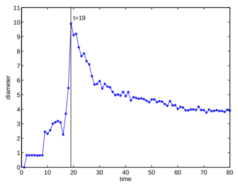
(b) *Arxiv* NLCCs



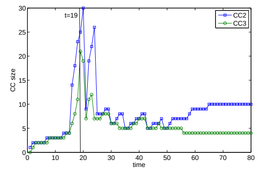
(a) *NIPS* Diam(t)



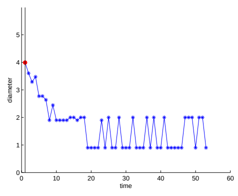
(b) *NIPS* NLCCs



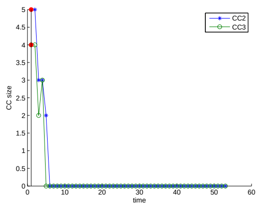
(a) *BlogNet* Diam(t)



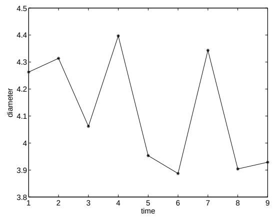
(b) *BlogNet* NLCCs



(a) *NetTraffic* Diam(t)



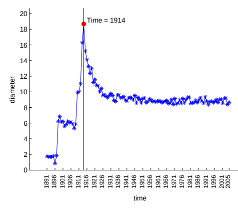
(b) *NetTraffic* NLCCs



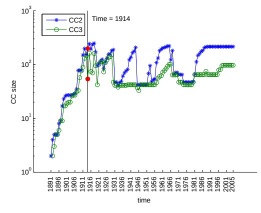
(a) *Oregon* Diam(t)

(b) *Oregon* NLCCs: N/A

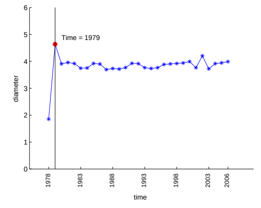
Figure 2: Properties of other unipartite networks. Diameter plot (left column), and NLCCs over time (right); vertical line marks the gelling point. All datasets exhibit an early gelling point, and stabilization of the NLCCs. *Oregon* has no NLCC plot since it consists of a single connected component, by its construction.



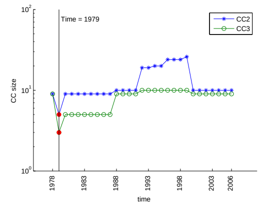
(a) *IMDB* Diam(t)



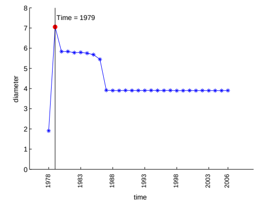
(b) *IMDB* NLCCs



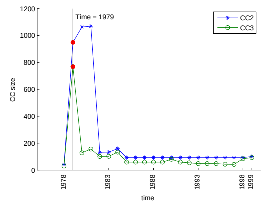
(c) *CampOrg* Diam(t)



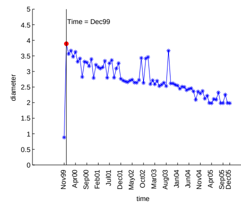
(d) *CampOrg* NLCCs



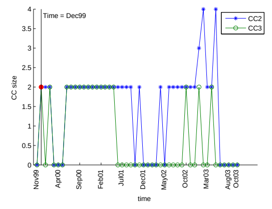
(e) *CampIndiv* Diam(t)



(f) *CampIndiv* NLCCs



(g) *Netflix* Diam(t)



(h) *Netflix* NLCCs

Figure 3: Properties of bipartite networks. Diameter plot (left column), and NLCCs over time (right), with vertical line marking the gelling point. Again, all datasets exhibit an early gelling point, and stabilization of the NLCCs. *Netflix* has strange behavior because it is masked (see text).

OBSERVATION 3 (WEIGHT POWER LAW (WPL)). *Let $E(t)$, $W(t)$ be the number of edges and total weight of a graph, at time t. They, they follow a power law*

$$W(t) = E(t)^w$$

where w is the weight *exponent. Power-laws also link the number of nodes $N(t)$, and the number of multi-edges $E_d(t)$, to $E(t)$, with exponents n and $dupE$, respectively.*

The weight exponent w ranges from 1.01 to 1.5 for the real graphs we have studied. The highest value corresponds to campaign donations: super-active organizations that support many campaigns also tend to spend even more money per campaign than the less active organizations. For bipartite graphs, we show the *nsrc*, *ndst* exponents for the source and destination nodes (which also follow power laws: $N_{src}(t) = E(t)^{nsrc}$ and similarly for $N_{dst}(t)$).

Fig. 5 shows all these quantities, versus $E(t)$, for several datasets. The plots are all in log-log scales, and straight lines fit well. We report the slopes in Table 2.

6.2 Snapshot Power Laws (SPL)

What about a static snapshot of a graph? If node i has out-degree out_i, what can we say about its out-weight $outw_i$? It turns out that there is a "fortification effect" here, too, resulting in more power laws, both for out-degrees/out-weights as well as for in-degrees/in-weights.

Specifically, at a given point in time, we plot the scatter-plot of the in/out weight versus the in/out degree, for all the nodes in the graph, at a given time snapshot. An example of such a plot is in Fig. 4 (c) and (d). Here, every point represents a node and the x and y coordinates are its degree and total weight, respectively. To achieve a good fit, we bucketize the x axis with logarithmic binning [21], and, for each bin, we compute the median y.

We observed that the median values of weights versus mid-points of the intervals follow a power law for all datasets studied. Formally, the "Snapshot Power Law" is:

Observation 4 (Snapshot Power Law (SPL)). *Consider the i-th node of a weighted graph, at time t, and let out_i, $outw_i$ be its out-degree and out-weight. Then*

$$outw_i \propto out_i^{ow}$$

where ow is the out-weight-exponent *of the SPL. Similarly, for the in-degree, with in-weight-exponent iw.*

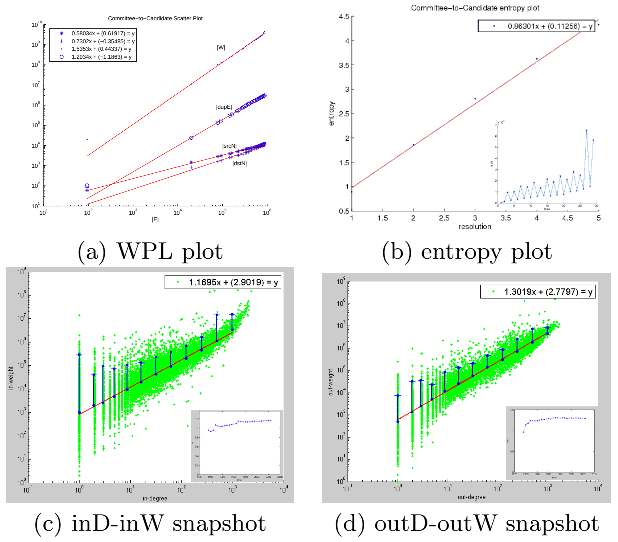
We studied the snapshot plots for several time-stamps (for brevity, we only report the slopes for the final timestamp in Table 2 for all the datasets we studied). We observed that SPL exponents of a graph over time remains almost constant. In Fig. 4 (c) ((d)), the inset plot shows how the $iw(ow)$ exponent changes over time (years) for the *CampOrg* dataset. We notice that iw and ow take values in the range [0.9-1.2] and [0.95-1.35], respectively. That is:

Observation 5. (Persistence of Snapshot Power Law) *The in- and out-exponents iw and ow of the SPL remain about constant, over time.*

Looking at Table 2, we observe that all SPL exponents are > 1, which imply a "*fortification effect*" with super-linear growth. The only exception is the *NetTraffic* dataset. This is explained because the number of nodes N has a limit that can not be exceeded (the total IP addresses at the institution of observation) (See WPL plot for *NetTraffic* in Fig. 5). Until N reaches that point, the slopes are iw=1.19 and ow=1.27 (again, showing a "fortification effect").

6.3 Weight additions

We tracked how much weight a graph puts on at each time interval and looking at the entropy plots, we observed that the weight additions over time show self-similarity. For those weighted graphs where the edge weight is defined as the number of reoccurences of that edge, the slope of the entropy plot was greater than 0.95, pointing out uniformity. On the other hand, for those graphs where weight is not in terms of multiple edges but some other feature of the dataset such as the amount of donations for the FEC dataset, we observed that weight additions are more bursty, the slope being as low as 0.6 for the Network Traffic dataset. Fig. 5



(a) WPL plot (b) entropy plot

(c) inD-inW snapshot (d) outD-outW snapshot

Figure 4: Weight properties of *CampOrg* donations: (a) shows all the power laws as well as the WPL; the slope in (b) is ~ 0.86 indicating bursty weight additions over time; (c) and (d) have slopes > 1 ("fortification effect"), that is, that the more campaigns an organization supports, the superlinearly-more money it donates, and similarly, the more donations a candidate gets, the more average amount-per-donation is received. Inset plots on (c) and (d) show iw and ow versus time. Note they are very stable over time.

(b) column shows the entropy plots for the weighted datasets we studied. ΔW values over time are also shown in insets at the bottom right corner of each figure.

Observation 6. (Bursty/Self-Similar Weight Additions) *In all our graphs, the addition of weight ($\Delta W(t)$) was self-similar, with fractal dimension ranging from ≈1 (smooth/uniform), down to 0.6 (bursty).*

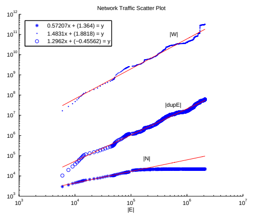
7. GENERATIVE MODEL

The next goal is to find a generative model that will produce a social network that obeys properties observed in this work as well as properties observed in previous work. We would like it to reproduce the following properties:

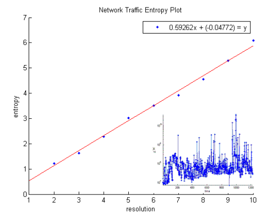
- Constant next-largest weakly connected component sizes.
- Densification power law
- Shrinking diameter (perhaps after a "gelling point")
- Power laws for in- and out-degree distribution

Moreover, we want an *emergent* generator, that will follow a simple, local behavior, out of which these global patterns will naturally emerge. Thus, we plan to have nodes arriving one at a time, and we want to design the method with which newcomers link to existing nodes, analogously to the 'preferential attachment' of Barabasi et. al. [3], but without the pitfalls of preferential attachment.

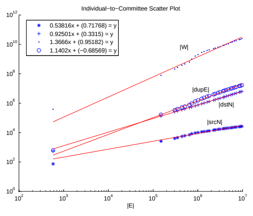
To achieve a long-tailed in-degree distribution, some form of preferential attachment will suffice. In order to even *have*



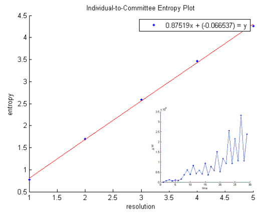
(a) *NetTraffic* WPLs



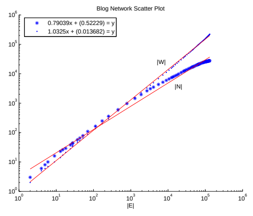
(b) *NetTraffic* entropy



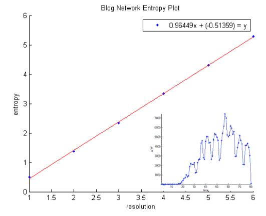
(a) *CampIndiv* WPLs



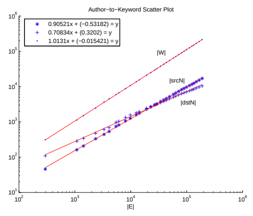
(b) *CampIndiv* entropy



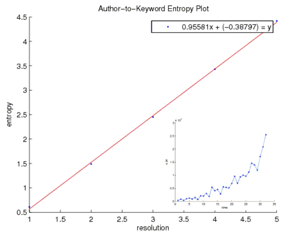
(a) *BlogNet* WPLs



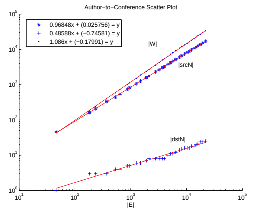
(b) *BlogNet* entropy



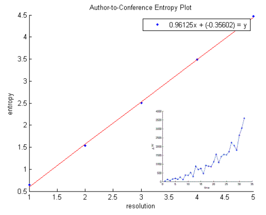
(a) *Auth-Key* WPLs



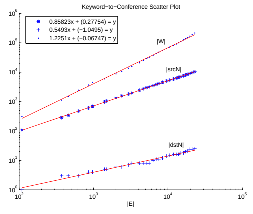
(b) *Auth-Key* entropy



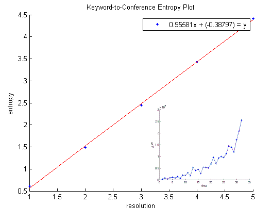
(a) *Auth-Conf* WPLs



(b) *Auth-Conf* entropy



(a) *Key-Conf* WPLs



(b) *Key-Conf* entropy

Figure 5: Properties of weighted networks. Left column: weight power laws for each graph studied, (W, E_d, N; vs E). The slopes for weight W and multi-edges E_d are above 1, indicating "fortification". Right column: entropy plots for weight addition. Slope away from 1 indicates burstiness (eg., 0.59 for *NetTraffic*) The inset plots show the corresponding time sequence ΔW versus time. Notice how bursty *NetTraffic* looks.

	w	$nsrc$	$ndst$	$dupE$	iw	ow	fd
CampOrg	1.53	0.58	0.73	1.29	1.16	1.30	0.86
CampIndiv	1.36	0.53	0.92	1.14	1.05	1.48	0.87
NetTraffic	1.48	0.57	NA	1.29	0.66	0.45	0.59
BlogNet	1.03	0.79	NA	NA	1.01	1.10	0.96
Auth-Key	1.01	0.90	0.70	NA	1.01	1.04	0.95
Auth-Conf	1.08	0.96	0.48	NA	1.04	1.81	0.96
Key-Conf	1.22	0.85	0.54	NA	1.26	2.14	0.95

Table 2: Power law exponents for all the weighted datasets we studied: The x-axis being the number of non-duplicate edges E, *w*: WPL exponent, *nsrc, ndst*: WPL exponent for source and destination nodes respectively (if the graph is unipartite, then *nsrc* is the number of all nodes), *dupE*: exponent for multi-edges, *iw, ow*: SPL exponents for indegree and outdegree of nodes, respectively. Exponents above 1 indicate fortification/superlinear growth. Last column, *fd*: slope of the entropy plots, or information fractal dimension. Lower fd means more burstiness.

disconnected components, we allow some newcomers to become 'bridges', that can link the GCC with an NLCC.

To achieve a power-law in the out-degree distribution, we *vary* one of the parameters of our model, so that it takes uniform values.

7.1 Generative "Butterfly" model

With these considerations, we present the following model, which we call the *Butterfly* as incoming nodes may behave as "social butterflies" by choosing more than one starting point, or "host", in their interactions; meeting nodes in the vicinity of the host, out-linking to some of them, and flying away. The model uses three parameters. The first, p_{link}, determines how often a link is formed between two nodes, and it is the same for all newcomers. The others, p_{host} and p_{step} are "friendliness" parameters: p_{step} decides whether the 'butterfly' will take one more step in its random walk; p_{host} is the probability it will take one more host. We set p_{step} to be *different* for each newcomer, uniformly distributed in the range, say [0,1]. We set p_{host} to be the same for all newcomers.[6] Expected number of hosts is $(1/(1-p_{host})-1)$ and expected number of steps per host chosen is $(1/(1-p_{step})-1)$.

In the model, nodes join the network one at a time. With probability p_{host}, an arriving node, denoted *current*, picks a host at random, and is assigned a p_{step} probability from a uniform distribution. After choosing the host, *current* travels in a random walk, recursively choosing at random one of the neighboring nodes (including both in- and out-links), taking each further step with probability p_{step}. Each time *current* visits (or re-visits) a node, it out-links to the visited node with p_{link} probability. Once the traveling stops, *current* returns to the starting state, choosing one more host with probability p_{host} and repeating, until no new hosts are chosen. Pseudocode for the model is shown in Fig. 6.

We choose $p_{host} = 0.5$ so the expected number of hosts is 1. However, once in a while, an arriving node chooses multiple hosts, allowing the possibility of two formerly disconnected components joining– which will reproduce the property of NLCCs remaining small.

[6]Letting p_{host} vary uniformly, also performed well empirically.

```
// generates a realistic looking graph
function butterfly
   global p_link = 0.3
   global p_host = 0.5
   global G = new_graph()
   for n = 1:N
       current=new_node()
       p_step = SampleUniform(0,1)
       G.add_node(current)
       while (SampleUniform() < p_host)
           host = G.random_node()
           visit(current, host)
   // May also return an undirected version,
   // to measure diameter
   return(G)

// input: a newcomer, and host node to visit
// effect: it updates G, with the new edges,
//   after current links to existing nodes
function visit(current, host)
    // with prob. p_link, link to the contact_node
    if (rand() < p_link)
       G.add_directed_edge(current, host)
    // with probability p_step, continue random walk
    if (SampleUniform() < current.p_step)
        next_visit = chooseRandom(host.neighbors())
        visit(current, next_visit)
```

Figure 6: Pseudocode for *Butterfly*.

7.2 Analysis

We find that choosing the parameters as defined in the above table, the results are remarkably similar to what is displayed in real graphs. Note that the model displays a stable or shrinking diameter, and that after a burning off period the second and third components demonstrate a threshold at which they do not grow further without joining the GCC.

THEOREM 1. *For a given host, the number of visits an arriving node forms follows power-law out-degree with exponent* -2.

PROOF. Taking p_{host} constant, the expected number of steps y that an arriving node will take is $\frac{1}{1-p_{step}} - 1$. ($1 - p_{step}$ is the probability of stopping traveling at any time point, so the number of steps taken before stopping follows a geometric distribution with mean $\frac{1}{1-p_{step}}$, and the number of visits is the number of steps before deciding to stop– the mean minus one.) If $p_{step} \sim Unif(0,1)$, we can do a transformation to find the distribution of the expected number of steps y [6]:

We represent $Y = g(X)$, and the distribution over X is uniform, $f_X(x) = 1$. Since the function $\frac{1}{x}$ is strictly monotone decreasing, then g has inverse $h = g^{-1}$, specifically $h(y) = \frac{1}{y}$. So we have

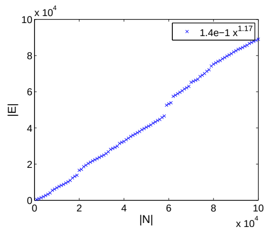
$$f_Y(y) \propto f_X(h(y)) * |\frac{dh(y)}{dy}| = h(y) * | - \frac{1}{x^2}| = f_Y(y) = x^{-2}$$

So, expected number of visits follows a power law with exponent -2. $\square$

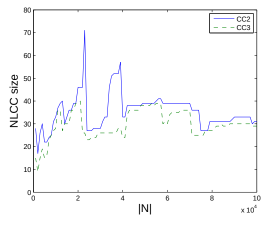
We believe that holding p_{link} constant, the degree distribution will follow a power law similar to the number of visits; and multi-edges and multiple hosts contribute a small amount to this factor, so empirical results perform well.

7.3 Empirical validation

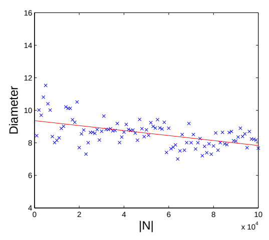
We simulated the model 10 times for $100,000$ nodes, with $p_{host} = 0.5$ and $p_{link} = 0.3$. One run's results are shown in Fig. 7. For each run the model exhibited power law in- and



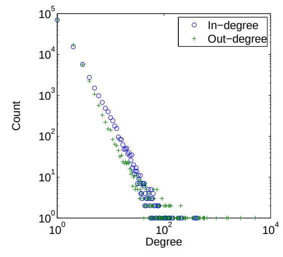
(a) N(t) vs E(t)



(b) CC sizes (log-lin)



(c) Diameter(N)



(d) Degree distribution

Figure 7: Results of proposed *Butterfly* model (p_{host}=0.5, p_{link}=0.3 p_{step} uniform. (a) Densification power law (exponent: 1.17) (b) Stabilizing NLCCs (between 20 and 50) (c) Small/shrinking diameter (d) power laws in the PDF of in- and out- degree distributions.

out-degree. Additionally, it displayed expected properties of the undirected graph– densification and stable NLCC sizes.

For p_{link} =0.3, the exponent had range $(1.03, 1.17)$ All occurring values of the exponent are within the range observed in real graphs, and have a least-squares fit of $R^2 > 0.99$ in log-log scales. Moreover, contrary to the *Forest Fire* method [17], our generator is robust, producing realistic-looking results for a wide range of parameter values (plots omitted for space). In contrast, small deviations from recommended parameter values in *Forest Fire* led to unrealistic densification exponents (either 1 or 2), and the model only produced a single GCC.

8. CONCLUSION

We believe that the *Butterfly* model and the observation of constant NLCC's will shed light upon other research in the area, such as a recent, counter-intuitive discovery [18]: the GCC of several real graphs has *no* good cuts, so graph partitioning and clustering algorithms cannot help identify communities because no clear communities exist.

The main contributions of this work are the following:

- **Patterns:** The discovery of several surprising patterns: early "gelling" point; power laws for weighted graphs (WPL, SPL); burstiness and self-similarity in the weight additions over time.
- **Generative Model:** Our proposed *Butterfly* model matches observations: it exhibits shrinking diameter, stabilizing NLCCs, and densification (with exponents comparable to real graphs). Furthermore it is intuitive, uses only local information and it is parsimonious, requiring only 3 parameters. It has further advantage over previous models in being robust, as parameters may be chosen over a large range of values and produce realistic graphs.

We ran extensive experiments on multiple, diverse, real graphs to identify common behaviors. We verified our observations on uni-partite, as well as bi-partite graphs, from blogs, publications, movie-actor networks, and many others. Our largest graph (*CampIndiv*) had 6 million nodes and 10 million edges, with annual time-stamps. Our datasets are in the public domain, and both our observations and our *Butterfly* model are easily reproducible.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grants No. IIS-0534205, IIS-0705359; and under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No.W-7405-ENG-48. Mary McGlohon was partially supported by a National Science Foundation Graduate Research Fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, or other funding parties.

The authors would like to thank Christopher Neff for help with the proofs.

9. REFERENCES

[1] R. Albert and A.-L. Barabasi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47, 2002.

[2] R. Albert, H. Jeong, and A.-L. Barabasi. Diameter of the world wide web. *Nature*, (401):130–131, 1999.

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

[4] R. Bell, Y. Koren, and C. Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–104, New York, NY, USA, 2007. ACM.

[5] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Link analysis ranking: algorithms, theory, and experiments. *ACM Trans. Inter. Tech.*, 5(1):231–297, 2005.

[6] G. Casella and R. L. Berger. *Statistical Inference*. Duxbury, 2002.

[7] D. Chakrabarti and C. Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, 38(1), 2006.

[8] S. Chakrabarti, B. E. Dom, S. R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Kleinberg. Mining the web's link structure. *Computer*, 32(8):60–67, 1999.

[9] Y. Chi, S. Zhu, X. Song, J. Tatemura, and B. L. Tseng. Structural and temporal analysis of the blogosphere through community factorization. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 163–172, New York, NY, USA, 2007. ACM.

[10] P. Erdos and A. Renyi. On the evolution of random graphs. *Publ. Math. Inst. Hungary. Acad. Sci.*, 5:17–61, 1960.

[11] A. Fabrikant, E. Koutsoupias, and C. H. Papadimitriou. Heuristically optimized trade-offs: A new paradigm for power laws in the internet (extended abstract), 2002.

[12] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. *SIGCOMM*, pages 251–262, Aug-Sept. 1999.

[13] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowedge Discover and Data Mining*, pages 611–617, New York, 2006.

[14] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Core algorithms in the clever system. *ACM Trans. Inter. Tech.*, 6(2):131–152, 2006.

[15] A. Lazarevic, L. Ertöz, V. Kumar, A. Ozgur, and J. Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the Third SIAM International Conference on Data Mining*, 2003.

[16] J. Leskovec, D. Chakrabarti, J. M. Kleinberg, and C. Faloutsos. Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. *PKDD*, pages 133–145, 2005.

[17] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 177–187, New York, NY, USA, 2005. ACM Press.

[18] J. Leskovec, K. Lang, A. Dasgupta, and M. Mahoney. Community structure in real graphs: The "negative dimensionality" paradox. In *International World Wide Web Conference*, 2008.

[19] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst. Cascading behavior in large blog graphs: Patterns and a model. In *Society of Applied and Industrial Mathematics: Data Mining*, 2007.

[20] S. Milgram. The small-world problem. *Psychology Today*, 2:60–67, 1967.

[21] M. E. J. Newman. Power laws, pareto distributions and zipf's law, December 2004.

[22] C. R. Palmer, P. B. Gibbons, and C. Faloutsos. Anf: A fast and scalable tool for data mining in massive graphs. In *SIGKDD*, Edmonton, AB, Canada, 2002.

[23] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 201–210, New York, NY, USA, 2007. ACM.

[24] S.-T. Park, D. M. Pennock, and C. L. Giles. Comparing static and dynamic measurements and models of the internet's as topology. In *INFOCOM*, 2004.

[25] D. M. Pennock, G. W. Flake, S. Lawrence, E. J. Glover, and C. L. Giles. Winners don't take all: Characterizing the competition for links on the web. *Proceedings of the National Academy of Sciences*, 99(8):5207–5211, 2002.

[26] S. Redner. Citation statistics from more than a century of physical review, Oct 2004.

[27] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing, 2002.

[28] M. Schroeder. *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*. W.H. Freeman and Company, New York, 1991.

[29] M. Wang, T. Madhyastha, N. H. Chang, S. Papadimitriou, and C. Faloutsos. Data mining meets performance evaluation: Fast algorithms for modeling bursty traffic. *ICDE*, Feb. 2002.

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