“Perfect” Power Law Graphs: Generation, Sampling, Construction, and Fitting

Jeremy Kepner

SIAM Annual Meeting, Minneapolis, July 9, 2012
Outline

• Introduction

• Sampling

• Sub-sampling

• Reuter’s Data

• Summary
Goals

• Develop a background model for graphs based on “perfect” power law

• Examine effects of sampling such a power law

• Develop techniques for comparing real data with a power law model
Can we construct a background model based on power law degree distribution?
• Graph represented as a rectangular sparse matrix
  – Can be undirected, multi-edged, self-loops, disconnected, hyper edges, …
• Out/in degree distributions are independent first order statistics
  – Only constraint: $\sum n(d_{out}) d_{out} = \sum n(d_{in}) d_{in} = M$
**Power Law Distribution Construction**

- **Perfect power law matlab code**

```matlab
function [di ni] = PPL(alpha,dmax,Nd)
logdi = (0:Nd) * log(dmax) / Nd;
di = unique(round(exp(logdi)));
logni = alpha * (log(dmax) - log(di));
ni = round(exp(logni));
```

- **Parameters**
  - `alpha` = slope
  - `dmax` = largest degree vertex
  - `Nd` = number of bins (before unique)

- **Simple algorithm naturally generates perfect power law**
- **Smooth transition from integer to logarithmic bins**
- **“Poor man’s” slope estimator:** \( \alpha = \log(n_1)/\log(d_{max}) \)
Power Law Edge Construction

- Power law vertex list matlab code

```matlab
function v = PowerLawEdges(di,ni);
A1 = sparse(1:numel(di),ni,di);
A2 = fliplr(cumsum(fliplr(A1),2));
[tmp tmp d] = find(A2);
A3 = sparse(1:numel(d),d,1);
A4 = fliplr(cumsum(fliplr(A3),2));
[v tmp tmp] = find(A4);
```

- Degree distribution independent of
  - Vertex labels
  - Edge pairing
  - Edge order

- Algorithm generates list of vertices corresponding to any distribution
- All other aspects of graph can be set based on desired properties
Fitting $\alpha$, N, M

- Power law model works for any
  - $\alpha > 0$, $d_{\text{max}} > 1$, $N_d > 1$

- Desire distribution that fits
  - $\alpha$, N, M

- Can invert formulas
  - $N = \sum_i n(d_i)$
  - $M = \sum_i n(d_i) d_i$

- Highly non-linear; requires a combination of
  - Exhaustive search, simulated annealing, and Broyden’s algorithm

- Given $\alpha$, N, M can solve for $N_d$ and $d_{\text{max}}$
- Not all combinations of $\alpha$, N, M are consistent with power law
Outline

• Introduction

• Sampling

• Sub-sampling

• Reuter’s Data

• Summary
Graph Construction Effects

- Generate a perfect power law NxN randomize adjacency matrix A
  - $\alpha = 1.3$, $d_{\text{max}} = 1000$, $N_d = 50$
  - $N = 18K$, $M = 84K$

- Make undirected, unweighted, with no self-loops
  
  $A = \text{triu}(A + A')$;
  $A = \text{double}(\text{logical}(A))$;
  $A = A - \text{diag}(\text{diag}(A))$;

- Graph theory best for undirected, unweighted graphs with no self-loops
- Often “clean up” real data to apply graph theory results
- Process mimics “bent broom” distribution seen in real data sets
### Power Law Recovery

#### Procedure

1. **Compute** $\alpha$, $N$, $M$ from measured data.

2. **Fit** perfect power law to these parameters.

3. **Rebin** measured data using perfect power law degree bins.

### Perfect power law fit to “cleaned up” graph can recover much of the shape of the original distribution.
Correlation Construction Effects

• Generate a perfect power law
  N\times N randomize incidence
  matrix E
  \( \alpha = 1.3, d_{\text{max}} = 1000, N_d = 50 \)
  \( N = 18K, M = 84K \)

• Make unweighted and use to
  form correlation matrix A with
  no self-loops

\[
E = \text{double}(\text{logical}(E));
A = \text{triu}(E' \times E);
A = A - \text{diag}(\text{diag}(A));
\]

• Correlation graph construction from incidence matrix results in a “bent
  broom” distribution that strongly resembles a power law
Power Law Lost

Procedure

- Compute $\alpha$, $N$, $M$ from measured
- Fit perfect power law to these parameters
- Rebin measured data using perfect power law degree bins

- Perfect power law fit to correlation shows non-power law shape
- Reveals “witches nose” distribution
Power Law Preserved

- In degree is power law
  - $\alpha = 1.3$, $d_{\text{max}} = 1000$, $N_d = 50$
  - $N = 18K$, $M = 84K$

- Out degree is constant
  - $N = 16K$, $M = 84K$
  - Edges/row = 5 (exactly)

- Make unweighted and use to form correlation matrix $A$ with no self-loops

- Uniform distribution on correlated dimension preserves power law shape
Edge Ordering: Densification

- Compute $M/N$ cumulatively and piecewise for 2 orderings
  - Linear
  - Random

- By definition $M/N$ goes from 1 to infinity for finite $N$

- Elimination of multi-edges reduces $M$ and causes $M/N$ to grow more slowly

- “Densification” is the observation that $M/N$ increases with $N$
  - Densification is a natural byproduct of randomly drawing edges from a power law distribution
  - Linear ordering has constant $M/N$
Edge Ordering: Power Law Exponent ($\alpha$)

- Compute $\alpha$ cumulatively and piecewise for 2 orderings
  - Linear
  - Random

- Edge ordering and sampling have large effect on the power law exponent

- Power law exponent is fundamental to distribution
- Strongly dependent on edge ordering and sample size
Outline

• Introduction

• Sampling

  • Sub-sampling

• Reuter’s Data

• Summary
Sub-Sampling Challenge

• Anomaly detection requires good estimates of background

• Traversing entire data sets to compute background counts is increasingly prohibitive
  – Can be done at ingest, but often is not

• Can background be accurately estimated from a sub-sample of the entire data set?
Sampling a Power Law

- Generate power law
- Select fraction of edges

**Whole distribution**

**1/40 sample**
Linear Degree Estimate

- Divide measured degree by fraction
- Accurate for high degree
- Overestimates low degree
- Can we do better?

Whole distribution

Linear estimate
Non-Linear Degree Estimate

- Assume power law input
- Create non-linear estimate
- Matches median degree
Outline

- Introduction
- Sampling
- Sub-sampling
  - Reuter’s Data
- Summary
Reuter’s Incidence Matrix

- Entities extracted from Reuter’s Corpus
- \( E(i,j) = \# \text{ times entity appeared in document} \)
- \( N_{\text{doc}} = 797677 \)
- \( N_{\text{ent}} = 47576 \)
- \( M = 6132286 \)
- Four entity classes with different statistics
  - LOCATION
  - ORGANIZATION
  - PERSON
  - TIME
- Fit power law model to each entity class
E(:,PERSON) Degree Distribution

<table>
<thead>
<tr>
<th>Type</th>
<th>M</th>
<th>N</th>
<th>M/N</th>
<th>α</th>
<th>M_{fit}</th>
<th>N_{fit}</th>
<th>M_{fit}/N_{fit}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document</td>
<td>299333</td>
<td>170069</td>
<td>1.76</td>
<td>1.92</td>
<td>302478</td>
<td>170066</td>
<td>1.78</td>
</tr>
<tr>
<td>Entity</td>
<td>299333</td>
<td>37191</td>
<td>8.05</td>
<td>1.21</td>
<td>299748</td>
<td>37449</td>
<td>8.00</td>
</tr>
</tbody>
</table>
E(:,PERSON)\(t\) \(\times\) E(:,PERSON)

**Procedure**

- Make unweighted and use to form correlation matrix A with no self-loops

\[
E = \text{double}(\text{logical}(E));
\]
\[
A = \text{triu}(E' \ast E);
\]
\[
A = A - \text{diag}(\text{diag}(A));
\]

- Perfect power law fit to correlation shows non-power law shape
- Reveals “witches nose” distribution
- Constant M/N consistent with sequential ordering of documents
Entity Densification

- Increasing M/N consistent with random ordering of entities
Document Power Law Exponent ($\alpha$)

- Increasing $\alpha$ consistent with sequential ordering of documents
Entity Power Law Exponent ($\alpha$)

- Decreasing $\alpha$ consistent with random ordering of entities
Summary

• Developed a background model for graphs based on “perfect” power law
  – Can be done via simple heuristic
  – Reproduces much of observed phenomena

• Examine effects of sampling such a power law
  – Lossy, non-linear transformation of graph construction mirrors many observed phenomena

• Traditional sampling approaches significantly overestimate the probability of low degree vertices
  – Assuming a power law distribution it is possible to construct a simple non-linear estimate that is more accurate

• Develop techniques for comparing real data with a power law model
  – Can fit perfect power-law to observed data
  – Provided binning for statistical tests
Acknowledgements

• Nicholas Arcolano
• Michelle Beard
• Nadya Bliss
• Bob Bond
• Matthew Schmidt
• Ben Miller
• Bill Arcand
• Bill Bergeron
• David Bestor

• Chansup Byun,
• Matt Hubbell
• Pete Michaleas
• Julie Mullen
• Andy Prout
• Albert Reuther
• Tony Rosa
• Charles Yee
Appendix
Sub-Sampling Formula

- $f =$ fraction of total edges sampled
- $n_1 =$ # of vertices of degree 1
- $d_{\text{max}} =$ maximum degree
- Allowed slope: $\ln(n_1)/\ln(d_{\text{max}}/f) < \alpha < \ln(n_1)/\ln(d_{\text{max}})$

- Cumulative distribution
  $$P(\alpha, d) = (f^{1-\alpha} d_{\text{max}}^\alpha / n_1) \sum_{i<d} i^{1-\alpha} e^{-fi}$$

- Find $\alpha^*$ such that $P(\alpha^*, \infty) = 1$
- Find $d_{50\%}$ such that $P(\alpha^*, d_{50\%}) = 1/2$
- Compute $K = 1/(1 + \ln(d_{50\%})/\ln(f))$

- Non-linear estimate of true degree of vertex $v$ from sample $d(v)$
  $$d(v) = \frac{d(v)}{f^{1-1/(K d(v))}}$$