PROBLEMS WITH INCOMPLETE NETWORKS: BIASES, SKewed RESULTS, AND SOLUTIONS

Sucheta Soundarajan
Syracuse University
susounda@syr.edu

Tina Eliassi-Rad
Northeastern University
tina@eliassi.org

Brian Gallagher
Lawrence Livermore Nat'I Lab
bgallagher@llnl.gov

Ali Pinar
Sandia Nat'I Laboratories
apinar@sandia.gov
Complex networks are ubiquitous

**Technological Networks**
- Internet
- NY State Power Grid

**Social Networks**
- Friendship
- HP Emails

**Information Networks**
- Map of Science

**Biological networks**
- Food Web
- Contagion of TB
Applications of complex networks

• Link analysis and web search
• Community detection
• Classification on networks
• Information maximization
• Social recommendation
• Epidemics
• Cascades
• ...

...
Properties of complex networks

- Size
- Density
- Average degree
- Degree distribution
- Average path length
- Diameter of a network
- Clustering coefficient
- Connectedness
- Node centrality
- …

High School Dating
(Bearman, Moody, and Stovel, 2004)
(Image by Mark Newman)
Properties of complex networks

- Size
- Density
- Average degree
- Degree distribution
- Average path length
- Diameter of a network
- Clustering coefficient
- Connectedness
- Node centrality
- …
Properties of complex networks

- Size
- Density
- Average degree
- Degree distribution
- Average path length
- Diameter of a network
- Clustering coefficient
- Connectedness
- Node centrality
- …
Properties of complex networks

- Size
- Density
- Average degree
- Degree distribution
- Average path length
- Diameter of a network
- Clustering coefficient
- Connectedness
- Node centrality

Three and a half degrees of separation

Some Facebook employees

- Mark Zuckerberg
  3.17 degrees of separation

- Sheryl Sandberg
  2.92 degrees of separation

The majority of the people on Facebook have averages between 2.9 and 4.2 degrees of separation. Figure 1 (below) shows the distribution of averages for each person.

Figure 1. Estimated average degrees of separation between all people on Facebook. The average person is connected to every other person by an average of 3.57 steps. The majority of people have an average between 3 and 4 steps.
Properties of complex networks

- Size
- Density
- Average degree
- Degree distribution
- Average path length
- Diameter of a network
- Clustering coefficient
- Connectedness
- Node centrality
- ...

The Anatomy of the Facebook Social Graph
Johan Ugander¹,², Brian Karrer¹,²,⁴, Lars Backstrom¹, Cameron Marlow¹†
¹ Facebook, Palo Alto, CA, USA
² Cornell University, Ithaca, NY, USA
³ University of Michigan, Ann Arbor, MI, USA
* These authors contributed equally to this work.
† Corresponding author: cameron@fb.com

Abstract

We study the structure of the social graph of active Facebook users, the largest social network ever analyzed. We compute numerous features of the graph including the number of users and friendships, the degree distribution, path lengths, clustering, and mixing patterns. Our results center around three main observations. First, we characterize the global structure of the graph, determining that the social network is nearly fully connected, with 99.91% of individuals belonging to a single large connected component, and we confirm the ‘six degrees of separation’ phenomenon on a global scale. Second, by studying the average local clustering coefficient and degeneracy of graph neighborhoods, we show that while the Facebook graph as a whole is clearly sparse, the graph neighborhoods of users contain surprisingly dense structure. Third, we characterize the assortativity patterns present in the graph by studying the basic demographic and network properties of users. We observe clear degree assortativity and characterize the extent to which ‘your friends have more friends than you’. Furthermore, we observe a strong effect of age on friendship preferences as well as a globally modular community structure driven by nationality, but we do not find any strong gender homophily. We compare our results with those from smaller social networks and find mostly, but not entirely, agreement on common structural network characteristics.

Properties of complex networks

- Size
- Density
- Average degree
- Degree distribution
- Average path length
- Diameter of a network
- Clustering coefficient
- Connectedness
- Node centrality
- Node Influence
- …
Incomplete networks

• Networked representations of real-world phenomena are often partially observed

• Acquiring more network data is often expensive and/or hard

• Even when your data is complete, you may not have the computational resources to examine all of the data
Roadmap

• Introduction -- Tina

• Part 1 -- Tina
  • Biases & Skewed Results

• Part 2 -- Sucheta
  • Enhancing incomplete Networks
    • MaxOutProbe
    • MaxReach
    • $\varepsilon$-WGX

• Wrap-up + Q&A -- Tina
PART 1: BIASES & SKEWED RESULTS
Partial observability

• Data comes in increments
  • Tweets
  • Wall posts
  • Packets
  • ...

• Data is expensive or difficult to collect
  • Protein-protein interactions determined experimentally
  • Can’t place a monitor every where on the Internet
Partial observability (cont.)

• Data comes in increments
• Data is expensive or difficult to collect
• Data is too big
Resort to sampling

• SDM 2015 Tutorial on *Methods and Applications of Network Sampling*
  • Tutors: Mohammad A. Hasan, Nesreen K. Ahmed, and Jennifer Neville
  • [http://www.siam.org/meetings/sdm15/sampling.php](http://www.siam.org/meetings/sdm15/sampling.php)
## Sampling

### Access Scenarios
- Complete access
- Crawling access
- Streaming access

### Objectives
- Sample to estimate properties of the original network at the micro-, meso-, and/or macro-levels
- Sample to obtain a small (preferably unbiased) portion of the original network
Sampling bias

• Which nodes are most likely to be present in the sample?
  • High degree nodes
  • Low degree nodes
  • …
Questions

• How can we accurately estimate
  • the degree distribution
  • the average degree
  • the global clustering coefficient
  • …
  
of the underlying network using the sample?
### Commonly used sampling strategies

<table>
<thead>
<tr>
<th><strong>Random</strong></th>
<th><strong>Crawl</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Uniform random node sampling (RNS)</td>
<td>• Random walk sampling (RWS)</td>
</tr>
<tr>
<td>• Uniform random edge sampling (RES)</td>
<td>• Breadth-first / snowball sampling (BFS)</td>
</tr>
</tbody>
</table>
Uniform random node sampling

1. Select $p$ fraction of nodes uniformly at random
2. Include all edges adjacent to the selected nodes

- Requires access to full list of nodes
- All nodes are equally likely to be selected
- High degree nodes are more likely to be observed as neighbors of selected nodes
Estimating degree distribution from a RNS

- Provides unbiased estimates for any nodal property
  - Average degree
  - Average of any attribute on the nodes
  - Degree distribution
- Disclaimer
  - If the sample is not the induced subgraph on the randomly selected nodes, then it is unlikely to capture the distribution’s tail well
Estimating clust. coeff. from a RNS

• Use the sample to estimate
  • $T = \#$ of triangles in underlying network
  • $W = \#$ of wedges in underlying network
• Clustering coefficient $C = T / W$
Estimating number of triangles from a RNS

- Probability that all three nodes are selected = $p^3$

- Probability that two of the three nodes are selected = $3p^2(1 - p)$

$T = \frac{T_s}{3p^2(1 - p) + p^3}$

Need at least two of these
Estimating the number of wedges from a RNS

- Probability that center node is selected = \( p \)
- Probability that endpoints, and not the center node, are selected = \( p^2(1 - p) \)
- \( W = \frac{W_s}{p^2(1 - p) + p} \)
Estimating clust. coeff. from a RNS

Putting it all together...

\[ \hat{C} = \frac{T^s}{W^s} \times \frac{(p(1 - p) + 1)}{3p(1 - p) + p^2} \]

which is an unbiased estimator for C
Uniform random edge sampling

1. Select $p$ fraction of edges uniformly at random from the network

- Requires access to full list of edges
- All edges are equally likely to be selected
- High degree nodes are more likely to be observed
Estimating degree distribution from a RES

- Degree distribution (and other statistics on nodes) will be biased towards high-degree nodes
- Edges statistics (such as assortativity) will be unbiased

Suppose a node $u$ has degree $d_u$ in the original network

- Its degree in the sample is given by a binomial distribution $Bin(d_u, p)$
  - Recall $p$ is the fraction of edges chosen uniformly at random
Estimating degree distribution from a RES (cont.)

Solve a least squares problem, where matrix coefficients come from binomial distribution

\[
\begin{bmatrix}
A(0, 1) & A(0, 2) & \ldots & A(0, n) \\
A(1, 1) & A(1, 2) & \ldots & A(1, n) \\
A(2, 1) & A(2, 2) & \ldots & A(2, n) \\
\vdots & \vdots & \ddots & \vdots \\
A(m, 0) & A(m, 1) & \ldots & A(m, n)
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix}
=
\begin{bmatrix}
y_0 \\
y_1 \\
\vdots \\
y_m
\end{bmatrix}
\]

\(A(i, j)\) = probability of node with true degree \(j\) having sample degree \(i\)

Estimated true degree counts

Observed degree counts
Estimating clust. coeff. from a RES

- Use the sample to estimate
  - \( T = \# \text{ of triangles in underlying network} \)
  - \( W = \# \text{ of wedges in underlying network} \)
- Clustering coefficient \( C = T / W \)
Estimating number of triangles from a RES

- Probability that all three edges are selected = $p^3$

- $T = \frac{T_s}{p}$
Estimating number of wedges from a RES

• Probability that both edges are selected = $p^2$

$W_s$

$W = \frac{W_s}{p^2}$
Estimating clust. coeff. from a RES

Putting it all together…

\[ \hat{C} = \frac{T^s}{W^s} \times \frac{1}{p} \]

which is an unbiased estimator for \( C \)
Random walk sampling

1. Begin with a random start node
2. At each step transition to a random neighbor of the current node

- Do not need access to full set of nodes or edges ahead of time (crawling)
- In RW’s stationary distribution, probability of observing a node $u$ is proportional to $d_u$
  - $P(u) = d_u / 2 |E|$
- High degree nodes are more likely to be present in the sample
Estimating structural properties from a RWS

If

the random walk is long enough to be approximated by its stationary distribution

then

degree distribution and clustering coefficient can be estimated using the same procedure as random edge sample
Breadth-first / snowball sampling

1. Begin with a random start node (or random start nodes)
2. Crawl the graph in a breadth-first fashion

- Do not need access to full set of nodes/edges ahead of time
- Vanilla BFS provides a complete “snapshot” of one area
- High degree nodes are more likely to be present in the sample
- Hard to estimate properties of underlying network from sample
Correcting biases in samples

• Exploration based sampling is biased toward high degree nodes
  • Can we modify the algorithms to ensure nodes are sampled uniformly at random?
  • Yes, uniform node sampling with Metropolis-Hastings method [Henzinger 2000]
• Sampling algorithms that selects nodes non-uniformly are biased when its comes to nodal statistics
  • Can we remove the sampling bias in nodal statistics by post-processing?
  • Yes, see Salganik & Heckathorn 2004

*From Hasan, Ahmed, and Neville, SDM’15 Tutorial.*
Roadmap

• Introduction -- Tina

• Part 1 -- Tina
  • Biases & Skewed Results

• Part 2 -- Sucheta
  • Enhancing incomplete Networks
    • MaxOutProbe
    • MaxReach
    • $\varepsilon$-WGX

• Wrap-up + Q&A -- Tina
PART 2: ENHANCING INCOMPLETE NETWORKS
Research question

Given limited resources, how should one gather more data to get the most bang for the buck?

Two approaches:

1. Model aware
2. Model agnostic
Given limited resources, how should one gather more data to get the most bang for the buck?

**Model Aware**

- Don’t gather more data
- Assume a graph model
- Use the incomplete network to fit a model of network structure
- Infer missing data
- A.k.a. network completion problem
The network completion problem

Given part of an adjacency matrix, infer the rest of the matrix

From Kim & Leskovec, SDM 2011

## Examples of network completion

<table>
<thead>
<tr>
<th>Hanneke &amp; Xing [AISTATS’09]</th>
<th>Kim &amp; Leskovec [SDM’11]</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Assume survey sample (nodes + neighbors)</td>
<td>• Assume Kronecker graph model</td>
</tr>
<tr>
<td>• Assume stochastic block model</td>
<td>• Cast problem in EM framework</td>
</tr>
<tr>
<td>• Assume that block memberships of surveyed nodes are known</td>
<td>• Given observed data, use a Metropolized Gibbs sampling method to estimate parameters of model and infer missing data</td>
</tr>
<tr>
<td>• Use observed data to estimate block connection probabilities</td>
<td>• Gives probability that two nodes are connected</td>
</tr>
</tbody>
</table>

The network inference problem

- Related to network completion
- Infer the network over which contagions propagate
- Lots of recent activity in this area
  - Nan Du, Le Song, Ming Yuan, Alex J. Smola: Learning Networks of Heterogeneous Influence. In NIPS 25, 2012: 2789--2797
Given limited resources, how should one gather more data to get the most bang for the buck?

**Model Aware**
- Don’t gather more data
- Assume a graph model
- Use the incomplete network to fit a model of network structure
- Infer missing data
- A.k.a. network completion problem

**Model Agnostic**
- Don’t assume a model
Model agnostic approaches may be worth considering when no model fits

Ongoing work with C. Seshadhri @ UCSC: Property testing in sparse graphs with realistic characteristics

(c) The CCDF of node degrees for each processing method and data source.

Inferred degree distributions of various methods vs. ground truth (red).*

None of the approaches provide confidences or guarantees on their results.

Given limited resources, how should one gather more data to get the most bang for the buck?

**Model Aware**
- Don’t gather more data
- Assume a graph model
- Use the incomplete network to fit a model of network structure
- Infer missing data
- A.k.a. network completion problem

**Model Agnostic**
- Don’t assume a model
- Infer missing data (e.g., link prediction) OR
- Collect additional data
Given limited resources, how should one gather more data to get the most bang for the buck?

**Model Aware**
- Don’t gather more data
- Assume a graph model
- Use the incomplete network to fit a model of network structure
- Infer missing data
- A.k.a. network completion problem

**Model Agnostic**
- Don’t assume a model
- Infer missing data (e.g., link prediction)
- OR
- Collect additional data

*Focus of this tutorial*
Issues to consider

 Goal

• Observe as many new nodes as possible
• Find triangles in the incomplete network
• Find links between “external” nodes
• ...

 Access Models

• Types of queries allowed
• Can ask for:
  • all the edges of a node
  • a random edge of a node
  • $k$ random edges of a nodes
  • all the communications between two nodes
  • ...

Roadmap for Part 2

- **MaxOutProbe**: A heuristic approach
  - **Goal**: Observe as many new nodes as possible
  - **Query**: Returns all the edges of a node

- **MaxReach**: A heuristic approach
  - **Goal**: Observe as many new nodes as possible
  - **Query**: Returns all edges, $k$ edges, or requested # edges

- **ε-WGX**: A multi-armed bandit approach
  - **Goal**: Observe as many new nodes as possible
  - **Query**: Returns a random edges of a node
MaxOutProbe
MaxOutProbe: Problem definition

• Given
  
  • An incomplete network $\hat{G}$ that is part of a larger, unseen network $G$
  
  • A probing budget $b$ in $\mathbb{R}$

• Goal
  
  • Select $b$ nodes from $\hat{G}$ that, when probed, bring as many new nodes as possible into $\hat{G}$

• Assumption
  
  • When a node is probed, all of its neighbors from $G$ are observed
Running example: $\hat{G}$
Running example

Which yellow nodes are adjacent to many green nodes?
Running example: Which yellow nodes are adjacent to many green nodes?
MaxOutProbe: Outline

1. Using $\hat{G}$, estimate each node $u$’s true degree $d_u$ in $G$

2. Estimate the number of neighbors $u$ has inside $\hat{G}$
   - Using $\hat{G}$, estimate the average clustering coefficient $C$ of $G$

3. Using #1 and #2, estimate the number of neighbors $u$ has outside $\hat{G}$
MaxOutProbe (cont.)

\[
d_{u}^{\text{out}} = d_{u} - d_{u}^{\text{in}} = d_{u} - (d_{u}^{\text{known}} + d_{u}^{\text{unknown}})
\]
Estimating degree of a node $d_u$

• Hypothesis
  • There is a scaling factor $s$ such that a node’s true degree can be approximated by $s$ times its observed degree

• How do we calculate $s$?
  • Sample a small number of high degree nodes from $\hat{G}$
  • Observe the ratio of their true degrees to their observed degrees
Estimating internal degrees

• Challenge
  • Given the structure of $\hat{G}$, how can we estimate the number of neighbors a node has inside $\hat{G}$?

• Observation: Nodes tend to cluster
  • If $u$ has many friends-of-friends inside $\hat{G}$, chances are $u$ is connected to some of them

• How many?
  • Use clustering coefficient to figure it out
  • Among the wedges, what fraction are closed triangles?
Running example

- $u$ has 4 friends-of-friends (red-lined yellow circles)
- $C =$ estimate of graph’s clustering coefficient
- Estimate that $u$ is connected to $4C$ of these nodes
Estimating $C$

- Reuse nodes probed during degree-estimation step

- When probed, what fraction of their friends-of-friends were they connected to?
Unbiased estimates

• MaxOutProbe obtains unbiased estimates if we know that
  • \( \hat{G} \) was produced by sampling nodes or edges uniformly at random from \( G \)

  and

  • the size of \( G \)

MaxOutProbe Experiments
## Datasets

<table>
<thead>
<tr>
<th>Network</th>
<th># of Nodes</th>
<th># of Edges</th>
<th>Transitivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twitter Retweets</td>
<td>40K</td>
<td>46K</td>
<td>0.03</td>
</tr>
<tr>
<td>Twitter Replies</td>
<td>261K</td>
<td>309K</td>
<td>0.002</td>
</tr>
<tr>
<td>Enron Emails</td>
<td>84K</td>
<td>326K</td>
<td>0.08</td>
</tr>
<tr>
<td>Yahoo! IM</td>
<td>100K</td>
<td>595K</td>
<td>0.08</td>
</tr>
<tr>
<td>Amazon Books</td>
<td>270K</td>
<td>741K</td>
<td>0.21</td>
</tr>
<tr>
<td>Youtube Videos</td>
<td>167K</td>
<td>1M</td>
<td>0.007</td>
</tr>
</tbody>
</table>
## Baseline & competing methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HighDeg</td>
<td>Select nodes with the highest degree.</td>
</tr>
<tr>
<td>LowDeg</td>
<td>Select nodes with the lowest degree.</td>
</tr>
<tr>
<td>HighDisp</td>
<td>Select nodes with the highest dispersion.</td>
</tr>
<tr>
<td>LowDisp</td>
<td>Select nodes with the lowest dispersion.</td>
</tr>
<tr>
<td>CrossCom</td>
<td>Select nodes with the highest fraction of neighbors outside of their community (detected by Louvain Method).</td>
</tr>
<tr>
<td>HighCC</td>
<td>Select nodes with the highest clustering coefficients.</td>
</tr>
<tr>
<td>LowCC</td>
<td>Select nodes with the lowest clustering coefficients.</td>
</tr>
<tr>
<td>Random</td>
<td>Randomly select nodes from the sample.</td>
</tr>
</tbody>
</table>
Experimental setup

- 20 trials
- Sample 10% of $G$’s edges using:
  - Random node sampling
  - Random edge sampling
  - Random walk
  - Random walk with jumps
- Run experiments at budgets $b$ in \{1\%, 2\%, 3\%, 4\%, 5\%\} of the # of nodes in each network
- Evaluate the quality of the enhanced graph by counting how many nodes it has
MaxOutProbe: Results

1. Compared to random probing, MaxOutProbe outperforms High Degree probing (the best baseline) by 4% - 36% on average.

2. Small improvements are because of tiny clustering coefficients.

[Graphs showing performance comparison for Enron email and Twitter replies]
MaxOutProbe: Summary

- Goal: Observe as many new nodes as possible
- Query: Returns all the edges of a node
- MaxOutProbe
  - Makes no assumptions about how the incomplete graph with generated or observed
  - Takes clustering coefficient into account
  - Improves performance over the best baseline algorithm (i.e., high-degree) by 4% to 36%
    - Improvement depends on $G$’s clustering coefficient
  - Tiny $C$, less improvement

MaxReach
MaxReach

• Similar problem definition as in MaxOutProbe

• Given
  • An incomplete network \( \hat{G} \) that is part of a larger, unseen network \( G \)
  • A probing budget \( b \) in \( \mathbb{R} \)

• Goal
  • Select \( b \) nodes from the evolving \( \hat{G} \) that, when probed, bring as many new nodes as possible into the current \( \hat{G} \)
MaxReach improves MaxOutProbe

1. Allows probing of new nodes as they are observed

2. Flexible access model
   - Example: Does not require all the edges of a probed node to be returned

3. More accurate degree and clustering coefficient estimates
MaxReach assumptions

- $\hat{G}$ was produced by random node or random edge sample; and we know which

- We know the size of $G$
  - # of nodes and # of edges
MaxReach improves degree estimates

- Suppose $\hat{G}$ was produced by sampling $p$ fraction of edges from $G$
- To estimate the degree distribution of $G$, solve the following least squares problem

$$
\begin{bmatrix}
B(0, 1) & B(0, 2) & \ldots & B(0, D) \\
B(1, 1) & B(1, 2) & \ldots & B(1, D) \\
B(2, 1) & B(2, 2) & \ldots & B(2, D) \\
B(\tilde{D}, 0) & B(\tilde{D}, 1) & \ldots & B(\tilde{D}, D)
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
\vdots \\
c_D
\end{bmatrix}
=
\begin{bmatrix}
c_0 \\
c_1 \\
\vdots \\
c_D
\end{bmatrix}
$$

$B(i, j) = \text{Prob. that node with degree } j \text{ in } G \text{ has degree } i \text{ in } \hat{G}$

Degree counts in $\hat{G}$

Degree counts in $G$ (to be estimated)
MaxReach improves degree estimates

- Our least squares problem is underdetermined
- Instead, use an EM-like iterative process to estimate the degree counts in $G$

**Iterative Estimation Process**

1. Initialize to uniform degree distribution
2. Estimate each node’s true degree
3. Update degree distribution
MaxReach improves degree estimates

• Calculate K-L divergence of the estimated degree distribution vs. the true distribution

• MaxReach performs $24-430 \times$ better than MaxOutProbe
MaxReach improves clust. coeff. estimates

- Clustering coefficient is related to degree
MaxReach improves clust. coeff. estimates

• Probability a wedge is preserved in $\hat{G} = p^2$
• Probability a triangle is preserved in $\hat{G} = p^3$
• Estimated CC = (Observed CC)/$\rho$
MaxReach improves clust. coeff. estimates

- MaxOutProbe estimates the \textit{global} average clust. coeff.
- MaxReach estimates a \textit{per-degree} average clust. coeff.
MaxReach estimates node statistics

1. Estimate each node $u$’s true degree in $G$ (i.e., $d_u$) by using
   - estimated degree distribution, and
   - $u$’s observed degree in $\hat{G}$

2. Estimate the number of neighbors $u$ has inside $\hat{G}$ (i.e., $d_u^{in}$) by using
   - estimated clustering coefficients of observed neighbors in $\hat{G}$

3. Estimate the number of neighbors $u$ has outside $\hat{G}$ (i.e., $d_u^{out}$) by using the estimates in #1 and #2
What is the access model?

1. All of a node’s edges?
   - Example: Facebook Graph API

2. $k$ of a node’s edges?
   - Example: Twitter API returns 5000 neighbors

3. A requested number of edges?
   - Assumption: There is a cost to initiate the request
MaxReach scores each node

• All of a node’s edges?
  • \( \text{Score}(u) = d_u^{\text{out}} \)

• \( k \) of a node’s edges?
  • \( \text{Score}(u) = \min\{k, d_u - d_u^{\text{known}}\} \times (d_u^{\text{out}} / (d_u - d_u^{\text{known}})) \)

• A requested number of edges, with a cost to initiate the request?
  • \( \text{Score}(u) = \max_k \left( \left( k d_u^{\text{out}} \right) / \left( (d_u - d_u^{\text{known}}) \cdot (rk + c) \right) \right) \)
    • \( k \) = \# of requested edges, such that \( k \leq d_u - d_u^{\text{known}} \)
    • \( c \) = request charge
    • \( r \) = cost per edges
MaxReach’s update step

• MaxReach updates node scores incrementally

• Allows us to make estimates for nodes as they are added to $\hat{G}$

• What is the expected degree of node $u$ given
  • its original observed degree in $\hat{G}$, and
  • the fact that its true degree $\geq$ its observed degree?

• Solution: Use Bayes’ Theorem and prior probabilities from $G$’s estimated degree distribution
MaxReach Experiments
Datasets

<table>
<thead>
<tr>
<th>Network</th>
<th># of Nodes</th>
<th># of Edges</th>
<th>Transitivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twitter Retweets</td>
<td>40K</td>
<td>46K</td>
<td>0.03</td>
</tr>
<tr>
<td>Twitter Replies</td>
<td>261K</td>
<td>309K</td>
<td>0.002</td>
</tr>
<tr>
<td>Enron Emails</td>
<td>84K</td>
<td>326K</td>
<td>0.08</td>
</tr>
<tr>
<td>Yahoo! IM</td>
<td>100K</td>
<td>595K</td>
<td>0.08</td>
</tr>
<tr>
<td>Amazon Books</td>
<td>270K</td>
<td>741K</td>
<td>0.21</td>
</tr>
<tr>
<td>DBLP</td>
<td>317K</td>
<td>1M</td>
<td>0.31</td>
</tr>
</tbody>
</table>
Experimental setup

• 10 trials
• Sample 10% of G’s edges using
  • Random node sampling
  • Random edge sampling
• Run experiments at various budgets
  • Budget depends on access model
• Evaluate quality of the enhanced graph by counting how many nodes it has
• Compare with adaptive versions of High Degree, Low Degree, and Random Probing
MaxReach: Results

On average, over all access models, MaxReach outperforms all baseline strategies.

![Probing on Enron Network](#)

- **MaxReach**
- **HighDegree**
- **LowDegree**
- **Random**

![Probing on DBLP Network](#)

- **MaxReach**
- **HighDegree**
- **LowDegree**
- **Random**

All-neighbor Probing

5-random-neighbor Probing
# MaxReach: Summary of Results

<table>
<thead>
<tr>
<th></th>
<th>All-neighbor access model</th>
<th>$k$-neighbor access model</th>
<th>Connection charge access model</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxReach outperforms</td>
<td>57-61%</td>
<td>9-59%</td>
<td>28-46%</td>
</tr>
<tr>
<td>Adaptive High Degree Probing by</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MaxReach: Summary

• Goal: Bring in as many nodes as possible

• MaxReach

  • Works under a variety of access models

  • Requires that the incomplete network was observed via random node or random edge sampling

  • Consistently outperforms other approaches when the goal is to increase # of nodes

ε-WGX: Problem definition

- **Adaptive Edge Probing (AEP)**
- **Given**
  - Incomplete network $\hat{G}$ that is part of a larger, unseen network $G$
  - Probing budget $b$ in $\mathbb{R}$
  - Reward function $R : (u, v) \rightarrow (r_u, r'_v)$, where $r_u, r'_v$ in $\mathbb{R}$
- **Goal**
  - Incrementally select $b$ nodes in $\hat{G}$ that, when probed, produce a graph $\hat{G}'$, where $\hat{G}$ in $\hat{G}'$ and $\hat{G}'$ maximizes cumulative reward
- **Assumption**
  - When a node is probed, one of its edges in $G$ is selected uniformly at random, including edges seen before
Challenges and questions for AEP

• No prior knowledge of how $\hat{G}$ was observed or generated

• Using only a node’s observed links in $\hat{G}$

• When to stop probing a node?

• Is there a general approach that works well across different reward functions and graphs from various domains?
Multi-armed bandits

- A multi-armed bandit is a tuple \( \langle \mathcal{A}, \mathcal{R} \rangle \)
- \( \mathcal{A} \) is a known set of \( m \) actions (or “arms”)
- \( \mathcal{R}^a(r) = P[r|a] \) is an unknown probability distribution over rewards
- At each step \( t \) the agent selects an action \( a_t \in \mathcal{A} \)
- The environment generates a reward \( r_t \sim \mathcal{R}^{a_t} \)
- The goal is to maximise cumulative reward \( \sum_{\tau=1}^{t} r_\tau \)

Slide Courtesy of David Silver, UCL
Exploration vs. exploitation

**Exploration**
- Pick an arm at random
- So, gather more information

**Exploitation**
- Pick the arm that maximizes reward given current information
  - So, make the best decision
MAB is a promising approach for AEP

- Can be used without background knowledge of network structure
- Can adapt to different reward function
- Is regularly providing the best performance for any given network and reward function
  - Disclaimer: based on our preliminary results
Some previous work on MAB with feedback graphs / side observations


- *Online Learning with Feedback Graphs: Beyond Bandits* by Noga Alon et al. 2015
Challenges in using MAB for AEP

• Changing rewards
  • Probability of getting a new edge decreases as a node is probed more
  • The graph itself can be changing
  • Complementarities: rewards depend on each other, even if those two nodes are not directly connected

• Short lifespan on bandits
  • Number of useful probes on any one node is likely to be small

• New arms get added
Graph complementarities

- Initially, $r^*(u) = \frac{1}{2}$ and $r^*(v) = \frac{1}{2}$
  - I.e., both $u$ and $v$ have half of their neighbors outside $\hat{G}$
- If we probe node $u$ and get edge $(u, w)$, $r^*(u) = 0$ and $r^*(v) = 0$
  - There is nothing left to learn for $u$
  - Because we have already seen node $w$, there is nothing left to learn for $v$ as well

$r^* = \text{true reward}$
$\varepsilon$-WGX: A nested bandit algorithm

- Outer Bandit
  - $\varepsilon_0$
  - $1-\varepsilon_0$

- Explore
  - 0.5
  - 0.5

- Inner Bandit
  - $1-\varepsilon_1$
  - $\varepsilon_1$

- Exploit
- Explore all nodes
- Explore unprobed nodes
Important aspects of $\varepsilon$-WGX

1. Different rewards for a node
   - One reward for when it is probed directly
   - Another reward for when it is observed as a neighbor

2. Probability of seeing a new edge from a node probe
Once a probe is made…

- $\epsilon$-WGX updates
  1. $r_u = \text{empirical mean reward for } u$
     which includes when $u$ was probed \textbf{and} when it was observed as a neighbor
  2. $p_u = \text{probability of seeing a new edge if } u \text{ is probed again}$
  3. $r_v$, if the observed neighbor $v$ was already in the observed network

- Expected reward of a node $u = p_u \times r_u$
Calculating $p_u$

- $p_u =$ probability of seeing a new edge when $u$ is probed
- Suppose node $u$ has been probed $k$ times with $w$ distinct neighbors and $h$ duplicates $\Rightarrow k = w + h$
- What is the estimated degree $d$ of node $u$?
- Same as predicting population size with random draws [Samuel 1968]

**MLE of $d$ is:**

\[
\hat{d} = \frac{w + h}{m\left(\frac{w}{k}\right)}, \text{ where } m(s) \text{ is the solution to } \frac{w}{k} = \frac{1 - e^{-m}}{m}
\]

- Assuming all edges are equally likely to be observed by a probe

\[
p_u = 1 - w/\hat{d}
\]

---

ε-WGX: Algorithmic Overview

- For each node $u$, keep track of the time-step that it was first observed, $T(u)$; each node $u$ in $\hat{G}$ has $T(u) = 0$
- While within budget $b$
  - Select a node $u$ for probing according to the nested bandit probabilities
  - Node $u$ is probed and edge $(u, v)$ is obtained
  - If $v$ is a newly observed node
    - Set $T(v) = \text{current time step}$
    - Add 1 to the cumulative sum for $r_u$ and increment its running count
  - Otherwise /* $v$ was observed previously */
    - If $T(v) > T(u)$ /* $v$ was observed after $u$ */
      - Add 1 to the cumulative sum for $r_u$
      - Add 0 to the cumulative sum for $r'_v$
    - Otherwise /* $v$ was observed before $u$ */
      - Add 0 to the cumulative sum for $r_u$
      - Add 1 to the cumulative sum for $r'_v$
      - Increment the running counts for $u$ and $v$
  - Update $r_u$, $r_v$ (if necessary), $p_u$
ε-WGX’s Regret

- Regret after $T$ time steps = expectation of the cumulative difference the estimated reward and the optimal reward
  - The optimal policy knows which arms are best to play and how the arms affect each other's rewards
- ε-WGX has linear regret (similar to ε-greedy)
  - ε-WGX explores a constant fraction of the time
  - ε-WGX assumes a static graph so the above does not hold as time goes to infinity
- Why? As time goes to infinity, regret goes to zero – i.e., you will get the complete graph
ε-WGX
Experiments
# Datasets

<table>
<thead>
<tr>
<th>Network</th>
<th># of Nodes</th>
<th># of Edges</th>
<th>G’s Avg. Clust. Coeff.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB Grad</td>
<td>0.5K</td>
<td>3.3K</td>
<td>0.48</td>
</tr>
<tr>
<td>FB Ugrad</td>
<td>1.2K</td>
<td>43K</td>
<td>0.30</td>
</tr>
<tr>
<td>Twitter Retweet</td>
<td>40K</td>
<td>46K</td>
<td>0.14</td>
</tr>
<tr>
<td>FB Social Circles</td>
<td>4K</td>
<td>88K</td>
<td>0.61</td>
</tr>
<tr>
<td>Twitter Replies</td>
<td>261K</td>
<td>309K</td>
<td>0.004</td>
</tr>
<tr>
<td>Enron Emails</td>
<td>84K</td>
<td>326K</td>
<td>0.15</td>
</tr>
<tr>
<td>Amazon Books</td>
<td>270K</td>
<td>741K</td>
<td>0.40</td>
</tr>
</tbody>
</table>
Baseline & competing methods

• $\epsilon$-greedy
• Upper Confidence Bound (UCB)
• High degree
• Low degree
• Random
Experimental setup: Incomplete graphs

- Sample 10% of G’s edges using:
  - Breadth-first crawl
  - Random edge sampling
  - Random walk
  - Random walk with jumps
Experimental setup: Probing budget

- $\hat{G} =$ incomplete graph
- $G =$ complete graph
- $K =$ # of edges in $G$ that are adjacent to at least one node in $\hat{G}$
- $M =$ $K -$ # of edges in $\hat{G}$
  - $M =$ # of edges that include nodes from $\hat{G}$, but have not yet been observed.
- Budget increments are the 100-quantiles in $[c_{min} \times M, c_{max} \times M]$
Experimental setup: Exploration parameters

• Largely empirical

  • Similar to $\epsilon$-greedy

• Outer bandit has two choices: prioritize unprobed nodes or not

  • Its exploration parameter set to 0.05

  • This parameter could be implemented with decay

• Inner bandit has many choices: which specific node to select

  • Its exploration parameter set to 0.3

  • This parameter should not decay
Evaluation

• 10 trials

• For each network & probing budget $b$
  
  • Calculate how much $\varepsilon$-WGX increased the # of nodes, divided by how much the comparative algorithm increased # of nodes

$$\frac{|\hat{V}'_{\varepsilon WGX} - \hat{V}|}{|\hat{V}'_{comp} - \hat{V}|}$$
**ε-WGX: Results**

- **Goal:** Observe as many new nodes as possible
- **Query:** Returns a random edges of a node

**How Often Does ε-WGX Beat Other Methods?**

<table>
<thead>
<tr>
<th>Incomplete network observed via</th>
<th>BFS</th>
<th>RandEdge</th>
<th>RW</th>
<th>RWJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>92%</td>
<td>76%</td>
<td>83%</td>
<td>80%</td>
</tr>
<tr>
<td>Low Degree (Best Baseline)</td>
<td>91%</td>
<td>86%</td>
<td>93%</td>
<td>88%</td>
</tr>
<tr>
<td>ε-Greedy</td>
<td>92%</td>
<td>86%</td>
<td>87%</td>
<td>73%</td>
</tr>
<tr>
<td>UCB</td>
<td>80%</td>
<td>83%</td>
<td>87%</td>
<td>84%</td>
</tr>
</tbody>
</table>
ε-WGX: Summary

- Adaptive Edge Probing (AEP) Problem
- MAB for AEP
- ε-WGX: nested MAB algorithm; model agnostic
- Regularly outperforms other approaches when the goal is to increase # of nodes
- Lots more work to be done
  - Collective classification of reward/regret
  - Dynamic graphs

Roadmap

• Introduction -- Tina

• Part 1 -- Tina
  • Biases & Skewed Results

• Part 2 -- Sucheta
  • Enhancing incomplete Networks
    • MaxOutProbe
    • MaxReach
    • ε-WGX

• Wrap-up + Q&A -- Tina
Wrap-up

• Beware your networked data is incomplete
• Mining can lead to skewed results
• Model-aware solution
  • Don’t gather data
  • Assume a model
  • Use the incomplete network to fit a model of network structure

• Model-agnostic solution
  Focus of this tutorial
  • Estimate local and/or global network statistics
  • Gather data based on your estimates
  • Repeat until out of budget
Future work

- Model aware + data gathering
- MAB for enhancing incomplete networks
- Handling incomplete dynamic graphs
- Treating the graph as a constrained network
- Full observability in parts of the graph but partial in other parts
- Semantics of a non-edge
- …
Thank you

- Slides and resources at [http://eliassi.org/sdm16tut.html](http://eliassi.org/sdm16tut.html)
- WIND’16: Workshop on Incomplete Networked Data
- Contact us at
  - susounda@syr.edu
  - tina@eliassi.org
- Supported by NSF, DTRA, DARPA, LLNL, and Sandia.