Measurements on (Complete) Graphs: The Power of Wedge and Diamond Sampling

Tamara G. Kolda
plus Grey Ballard, Todd Plantenga, Ali Pinar, C. Seshadhri

Workshop on Incomplete Network Data
Sandia National Labs
Livermore, CA
March 22, 2016
Error Bounds for Sampling

Given a huge bin of colored balls. Let $\mu$ be the unknown proportion of red balls.

Random Variable

$$X_i = \begin{cases} 
1 & \text{if } i\text{th draw is red} \\
0 & \text{otherwise}
\end{cases}$$

Sample Mean

$$\bar{X} = \frac{1}{k} \sum_{i=1}^{k} X_i$$

But how far away is the sample mean from the mean after $k$ samples?

Hoeffding Inequality (1963)

For $\epsilon \in (0, 1)$,

$$\text{Prob} \left\{ |\bar{X} - \mu| \geq \epsilon \right\} \leq \delta$$

where $\delta \equiv 2 \exp(-2k\epsilon^2)$

Probabilistic error bound:

error $\epsilon$ and confidence $(1-\delta)$
Clustering Coefficients & Triangles

\[ d_i = \text{degree of node } i \]

\[ t_i = \text{triangles involving node } i \]

Node clustering coefficient: \( c_i = t_i / \binom{d_i}{2} \)

\[ c_B = 1 \]

\[ c_F = \frac{1}{6} \]

\[ c_G = 0 \]

Degree-\(d\) clustering coefficient: \( c_d = \text{mean} \left\{ c_i \mid d_i = d \right\} \)

Global clustering coefficient: \[ c = \frac{\sum_i t_i}{\sum_i \binom{d_i}{2}} = \frac{3 \times \text{total number of triangles}}{\text{total number of wedges}} \]
c = probability that a uniform random wedge is closed

**Enumeration**: Find every wedge. Check if each is closed.
\[ c = \frac{\text{# closed wedges}}{\text{# wedges}} \]

**Sampling**: Sample a few wedges (uniformly). Check if each is closed.
\[ c \approx \frac{\# \text{ closed sampled wedges}}{\# \text{ sampled wedges}} \]
Uniform Wedge Sampling

\[ w_j = \binom{d_j}{2} = \frac{d_j(d_j - 1)}{2} = \text{number of wedges centered at node} \ j \]

\[ w = \sum_j w_j = \text{total number of wedges} \]

To pick random wedge \((i, j, k)\):

1. Choose \(j \propto w_j / w\)

2. Choose \(i, k \in \mathcal{N}(j)\) such that \(i \neq k\)

\[
\text{Prob(wedge \((i, j, k)\)) = Prob(vertex \ j) \cdot Prob(wedge \((i, j, k)\)|center \ j) = \frac{w_j}{w} \cdot \frac{1}{w_j} = \frac{1}{w}}
\]
Sampling is Faster than Enumeration

<table>
<thead>
<tr>
<th>Graph</th>
<th>Nodes</th>
<th>Edges</th>
<th>Wdgs</th>
<th>Trngls</th>
</tr>
</thead>
<tbody>
<tr>
<td>amazon0312</td>
<td>401K</td>
<td>2350K</td>
<td>69M</td>
<td>3686K</td>
</tr>
<tr>
<td>amazon0505</td>
<td>410K</td>
<td>2439K</td>
<td>73M</td>
<td>3951K</td>
</tr>
<tr>
<td>amazon0601</td>
<td>403K</td>
<td>2443K</td>
<td>72M</td>
<td>3987K</td>
</tr>
<tr>
<td>as-skitter</td>
<td>1696K</td>
<td>11095K</td>
<td>16022M</td>
<td>28770K</td>
</tr>
<tr>
<td>cit-Patents</td>
<td>3775K</td>
<td>16519K</td>
<td>336M</td>
<td>7515K</td>
</tr>
<tr>
<td>roadNet-CA</td>
<td>1965K</td>
<td>2767K</td>
<td>6M</td>
<td>121K</td>
</tr>
<tr>
<td>web-BerkStan</td>
<td>685K</td>
<td>6649K</td>
<td>27983M</td>
<td>64691K</td>
</tr>
<tr>
<td>web-Googel</td>
<td>876K</td>
<td>4322K</td>
<td>727M</td>
<td>13392K</td>
</tr>
<tr>
<td>web-Stanford</td>
<td>282K</td>
<td>1993K</td>
<td>3944M</td>
<td>11329K</td>
</tr>
<tr>
<td>wiki-Talk</td>
<td>2394K</td>
<td>4660K</td>
<td>12594M</td>
<td>9204K</td>
</tr>
<tr>
<td>youtube</td>
<td>1158K</td>
<td>2990K</td>
<td>1474M</td>
<td>3057K</td>
</tr>
<tr>
<td>flickr</td>
<td>1861K</td>
<td>15555K</td>
<td>14670M</td>
<td>548659K</td>
</tr>
<tr>
<td>livejournal</td>
<td>5284K</td>
<td>48710K</td>
<td>7519M</td>
<td>310877K</td>
</tr>
</tbody>
</table>

1000X speed-up on average!
Accuracy is Better than Predicted

Error is generally smaller than Hoeffding conservatively predicts

At 99.9% confidence ($\delta = 0.001$), the error bounds are...

\[
\begin{align*}
    k = 2000 & \Rightarrow \epsilon = 0.043 \\
    k = 8000 & \Rightarrow \epsilon = 0.022 \\
    k = 32000 & \Rightarrow \epsilon = 0.011
\end{align*}
\]
Map-Reduce for Wedge Samples

- Calculate degree distribution
  - Calc. degree per vertex
  - Calc. degree distribution and num. wedges (per bin)

- Choose sample of uniform random wedges
  - Pick a few wedge centers
  - Generate random wedges

- Check closure of each sampled wedge
  - Check each wedge for closure

- Collect results

$$C \approx \frac{\# \text{ closed sample wedges}}{\# \text{ sample wedges}}$$
Large Data Sets and Machines

- 5 real-world networks
  - Source: Laboratory for Web Algorithms (Italy)
  - Largest: 132M nodes, 4.6B edges
  - Observe: # wedges $\gg$ # edges!

- Compute Servers
  - Big Memory Server: SGI Altix UV 10
    - 32 cores (4 Xeon 8-core 2.0GHz processors)
    - 512 GB DDR3 memory
  - Distributed Server: 32-Node Hadoop Cluster
    - 32 x Intel 4-Core i7 930 2.8GHz CPU = 128 cores
    - 32 x 12GB = 384GB memory
    - 32 x 4 2TB SATA disks = 256TB disk storage

Mar 22, 2106
Wedge Sampling for BIG Graphs

- 32-node Hadoop cluster results using wedge sampling
  - Logarithmic bins
  - 2000 samples per bin
- “BIG” graphs benefit from Hadoop
  - Merely “Big” graphs don’t require Hadoop – just shown as examples
- Compare twitter times
  - Sampling: 10 mins on 32-node Hadoop cluster
  - Enumeration: 483 mins on 1636-node Hadoop cluster
    - Suri & Vassilvitskii, 2011
  - Enumeration: 180 mins on 32-core SGI, using 128GB RAM
    - by Jon Berry, 2013
- No comparisons for uk-union due to its size
twitter-2010: 41M nodes, 1.2B undirected edges, 123T wedges, 31B triangles

Nodes are users. Connected if one follows the other.

32-node Hadoop Cluster
37 bins x 380 samples/bin
Degree per Vertex = 171s
Degree Dist. = 31s
Sample Wedges = 121s
Check Wedges = 91s
Post-processing = 110s
Total: 530s (=9m)

Compare (Suri, Vassilvitskii, 2011):
483m on 1636-node Hadoop Cluster for full enumeration.

Data Source: Laboratory for Web Algorithms http://law.di.unimi.it/datasets.php
Similar Applications of Wedge Sampling

- Clustering coefficient *per degree*
  - Require center to have specified degree
- Triangles per degree
  - Need to adjust the counts based on the number of vertices with the *same degree* in the sampled wedge
- Directed triangle counts
  - Multiple occurrences of the same wedge type causes counting the same triangle multiple times.

Mar 22, 2106
Kolda - WIND
Problem: Maximum All-pairs Dot-product (MAD) Search

Given two sets of vectors in $\mathbb{R}^d$: $A = \{ a_1, \ldots, a_m \}$ and $B = \{ b_1, \ldots, b_n \}$

Find top-$t$ pairs $(i, j)$ that maximize: $| a_i \cdot b_j |$

Equivalent to finding the maximum-magnitude entries in the matrix product:

$$C = A^T B$$

$$c_{ij} = a_i^T b_j$$

$$m \times n$$

$$m \times d$$

$$d \times n$$
If there is a "wedge" \((i,k,j)\), then \(c_{ij} \geq 1\)

\[
prob(\text{wedge } (i,*,j)) \propto c_{ij}
\]
Pair of Wedges = Diamond

Diamond = Two Intersecting Wedges

If there is a “diamond” \((k', i, k, j)\), then \(c_{ij} \geq 2\)

\[ c_{ij}^2 = \# \text{ diamonds between nodes } i \text{ and } j \]

\[ \text{Prob( diamond } (\ast, i, \ast, j) \text{ ) } \propto c_{ij}^2 \]

Diamond: \((k', i, k, j)\)

Idea: Randomly select diamonds. The pairs \((i, j)\) that are selected the most times will correspond to the highest entries of \(C\) with high probability.
Enumerating all diamonds is too expensive. Instead, we need an inexpensive way to sample.

**Step 1:** Choose edge \((i,k) \propto w_{ik}\)

**Step 2:** Choose neighbor of \(k\)

**Step 3:** Choose neighbor of \(i\)

**Step 4:** Check for edge \((k',j)\) in \(B\)

Number of three-paths centered at edge \((i,k):\)

\[ w_{ki} = \deg_i^A \deg_k^B \]

Sample \(j\) from \(N_k^B\)

Sample \(k'\) from \(N_i^A\)

If edges exists, diamond complete:

\[ x_{ij} \leftarrow x_{ij} + 1 \]

Three-path sampling: Jha et al. WWW 2015
Sampling Random Diamonds for General Inputs

Step 1: Choose edge \((i, k) \propto w_{ik}\)

\[ w_{ki} = \frac{|a_{ki}| \|a_i\|_1 \|b_{k*}\|_1}{\|b_{kj}\|} \]

Step 2: Choose neighbor of \(k\)

Sample \(j\) with probability \(|b_{kj}| / \|b_{k*}\|_1\)

Step 3: Choose neighbor of \(i\)

Sample \(k'\) with probability \(|a_{k'i}| / \|a_i\|_1\)

Step 4: Check for edge \((k', j)\) in \(B\)

\[ x_{ij} \leftarrow x_{ij} + \text{sgn}(a_{ki} b_{kj} a_{k'i}) b_{k'j} \]
Quick Look: Up to 100X Speed-up Compared to Direct Computation

- Matrices
  - $A = d \times m$, $B = d \times n$

- Compute
  - $C = A^T B = m \times n$ matrix
  - Expensive in memory & time

- Memory workaround
  - Compute one result column at a time and *use a priority queue to hold the largest entries*: $c_j = A^T b_j$ for $j = 1, \ldots, n$
  - Exploit symmetry for $B = A$

- CSPARSE from Tim Davis
  - Modified cs_multiply function

---

$A = B = \text{user-song listens}$

$d = 1\text{M users}$, $m = n = 350\text{K songs}$

nnz($A$) = 48M

$C = A^T A = \text{song similarity}$

max samples = $10^7$
Results: Million Song

- Song plays by users
- $A$ is sparse and $B = A$
  - $m = 384,546$ songs
  - $n = d = 1,019,318$ users
  - nnz$(A) = 48,373,586$
- Max entry in $C$: $6.1 \times 10^6$
- # Wedges = $1.4 \times 10^{15}$
- # Samples = $O(10^1)$
- Closure rate: 15%

- **Speed-up of 5-100X**
- **Top song pair:** “Undo” by Bjork and “Revelry” by Kings of Leon
Motivating Example: Free Samples!

Book seller wants to increase reviews on its site.

\[ a_{ki} = \begin{cases} 
1 & \text{if person } i \text{ reviewed book } k \\
0 & \text{otherwise}
\end{cases} \]

\[ b_{kj} = \begin{cases} 
1 & \text{if book } j \text{ similar to book } k \\
0 & \text{otherwise}
\end{cases} \]

Send free books to users who have written reviews of similar books.

\[ c_{ij} = \# \text{ books reviewed by person } i \text{ that are similar to book } j \]
Results: Amazon Kindle

- Product recommendations
- $A$ is sparse reviewer-by-book rating matrix and $B$ is sparse book similarity matrix
  - $m = 1,406,916$ reviewers
  - $n = d = 430,532$ books
  - $\text{nnz}(A) = 3,205,546$
  - $\text{nnz}(B) = 11,012,558$
- Max entry in C: $3.2 \times 10^3$
- # Wedges = $1.2 \times 10^{12}$
- # Samples = $O(10^{12})$
- Closure rate: 1.4%

![Recall: amazon-kindle](image)

![Timing: amazon-kindle](image)
Example Reviewer-Book Recommendation

Recommended Book

Recommended User ID = 1317513

Previous Reviews Include…

1. Stay by Riley Hart
2. Flare-Up by Laura Harner
3. In Your Eyes by Cardeno C.
4. The Mating of Michael by Eli Easton
Comparison to Wedge Sampling

Recall: as-Skitter

Timing: as-Skitter

Wedge Sampling: Cohen & Lewis, 1997, 1999
Conclusions & References

- Sampling is a powerful tool for understanding large-scale graphs
  - Agreement in expectation
  - Error as a function of number of samples
  - Amenable to parallelism
- Incomplete/uncertain data?
  - Results are probability based
  - Maybe hope if we can estimate the probability of graph structures
  - Tensor/matrix completion?