

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

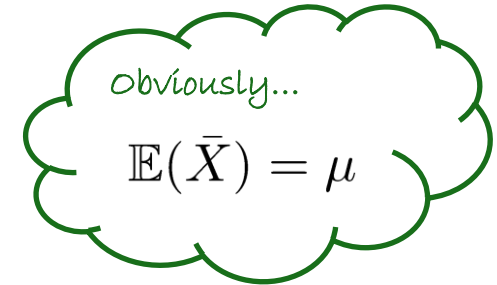
Given a *huge* bin of colored balls. Let μ be the unknown proportion of red balls.

Random Variable

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

Sample Mean

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

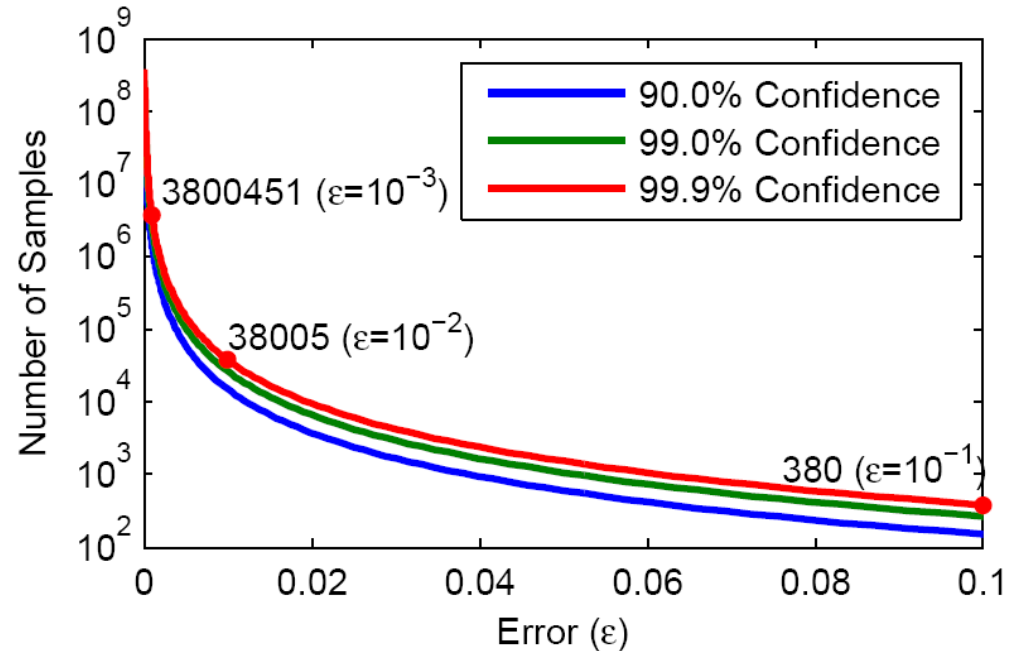


But how far away is the sample mean from the mean after k samples?

Hoeffding Inequality (1963)

For $\epsilon \in (0, 1)$,
 $\text{Prob} \{ |\bar{X} - \mu| \geq \epsilon \} \leq \delta$
 where $\delta \equiv 2 \exp(-2k\epsilon^2)$

*Probabilistic error bound:
 error ϵ and confidence $(1-\delta)$*

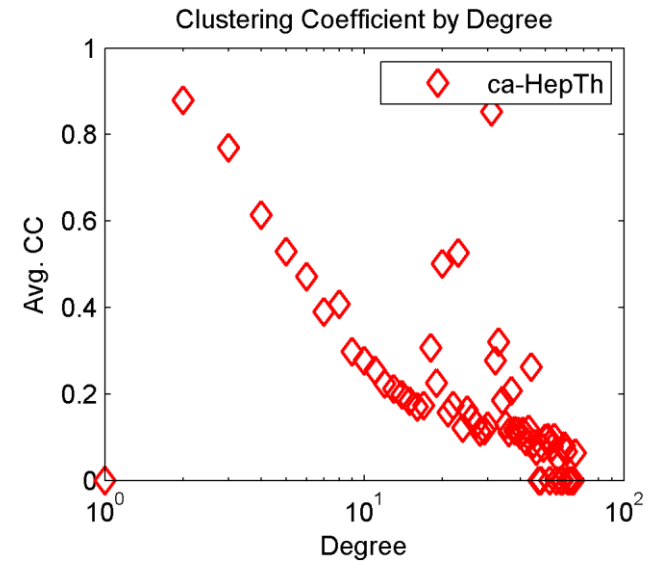
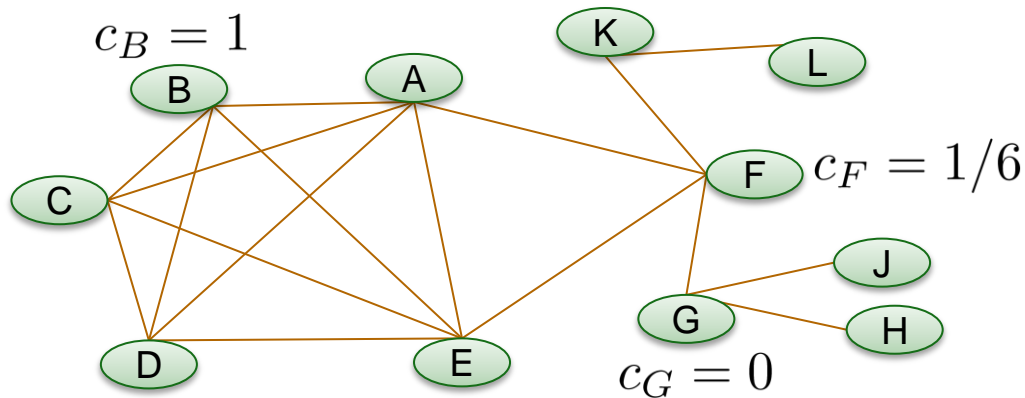


Clustering Coefficients & Triangles

d_i = degree of node i

t_i = triangles involving node i

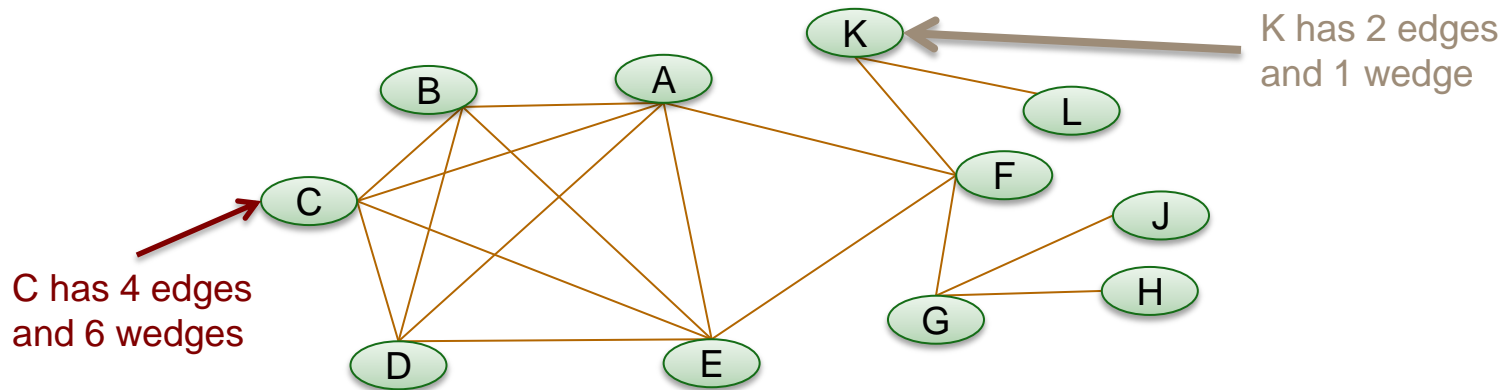
Node clustering coefficient: $c_i = t_i / \binom{d_i}{2}$



Degree- d clustering coefficient: $c_d = \text{mean} \{c_i | d_i = d\}$

Global clustering coefficient: $c = \frac{\sum_i t_i}{\sum_i \binom{d_i}{2}} = \frac{3 \times \text{total number of triangles}}{\text{total number of wedges}}$

Uniform Wedge Sampling



$$w_j = \binom{d_j}{2} = \frac{d_j(d_j-1)}{2} = \text{number of wedges centered at node } j$$
$$w = \sum_j w_j = \text{total number of wedges}$$

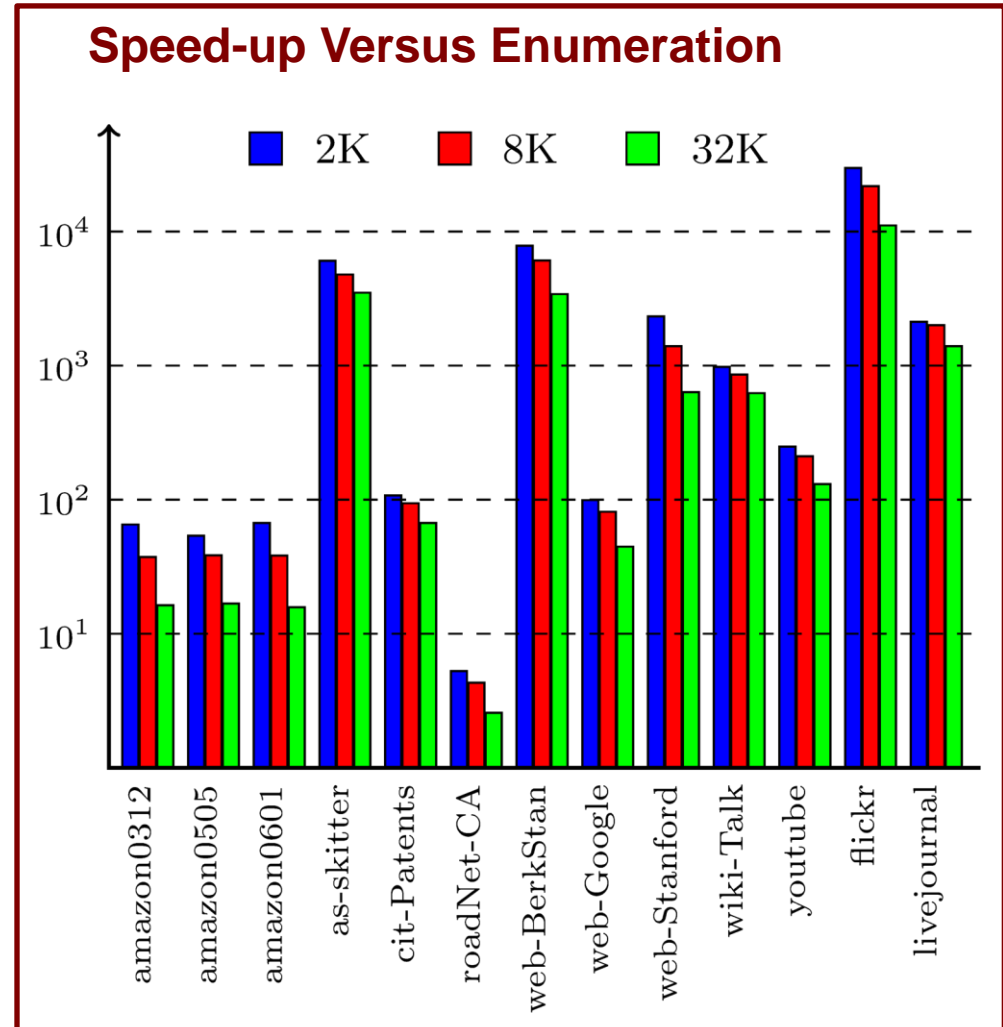
To pick random wedge (i, j, k) :

1. Choose $j \propto w_j/w$
2. Choose $i, k \in \mathcal{N}(j)$ such that $i \neq k$

$$\text{Prob}(\text{wedge } (i, j, k)) = \text{Prob}(\text{vertex } j) \cdot \text{Prob}(\text{wedge } (i, j, k) | \text{center } j) = \frac{w_j}{w} \cdot \frac{1}{w_j} = \frac{1}{w}$$

Sampling is Faster than Enumeration

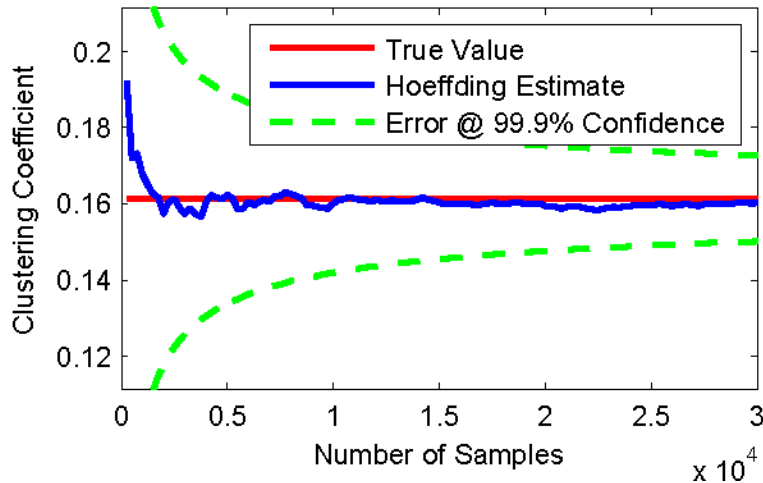
Graph	Nodes	Edges	Wdgs	Trngls
amazon0312	401K	2350K	69M	3686K
amazon0505	410K	2439K	73M	3951K
amazon0601	403K	2443K	72M	3987K
as-skitter	1696K	11095K	16022M	28770K
cit-Patents	3775K	16519K	336M	7515K
roadNet-CA	1965K	2767K	6M	121K
web-BerkStan	685K	6649K	27983M	64691K
web-Google	876K	4322K	727M	13392K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



1000X speed-up on average!

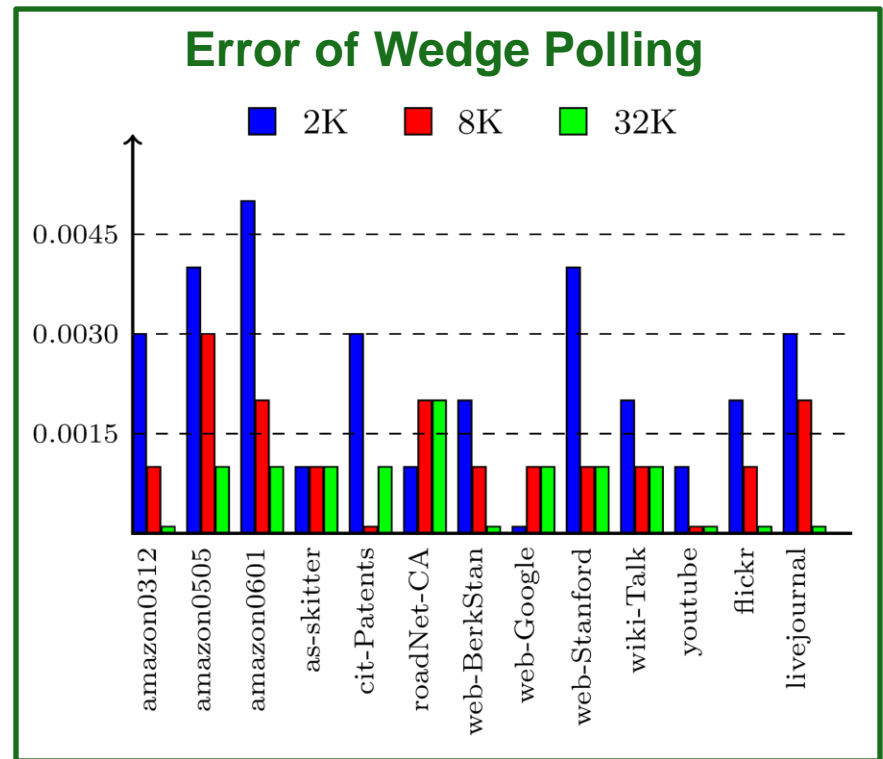
Accuracy is Better than Predicted

Error is generally smaller than Hoeffding conservatively predicts



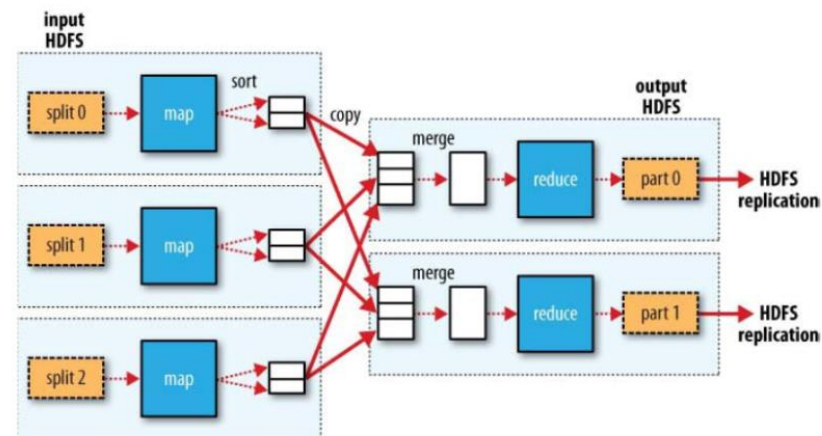
At 99.9% confidence ($\delta = 0.001$), the error bounds are...

$$\begin{aligned} k = 2000 &\Rightarrow \epsilon = 0.043 \\ k = 8000 &\Rightarrow \epsilon = 0.022 \\ k = 32000 &\Rightarrow \epsilon = 0.011 \end{aligned}$$

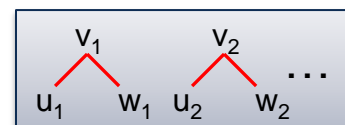


Map-Reduce for Wedge Samples

- 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**

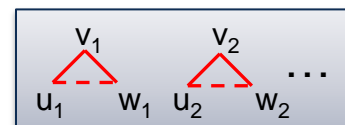


- Check closure of each sampled wedge
 - Check each wedge for closure**



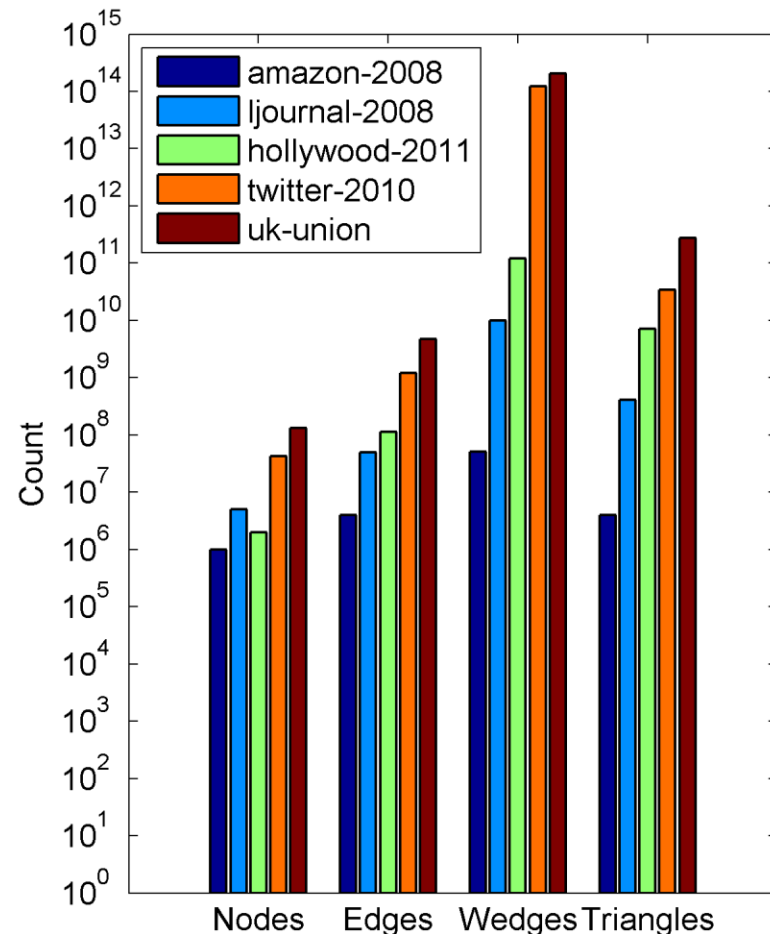
- Collect results

$$c \approx \frac{\# \text{ closed sample wedges}}{\# \text{ sample wedges}}$$



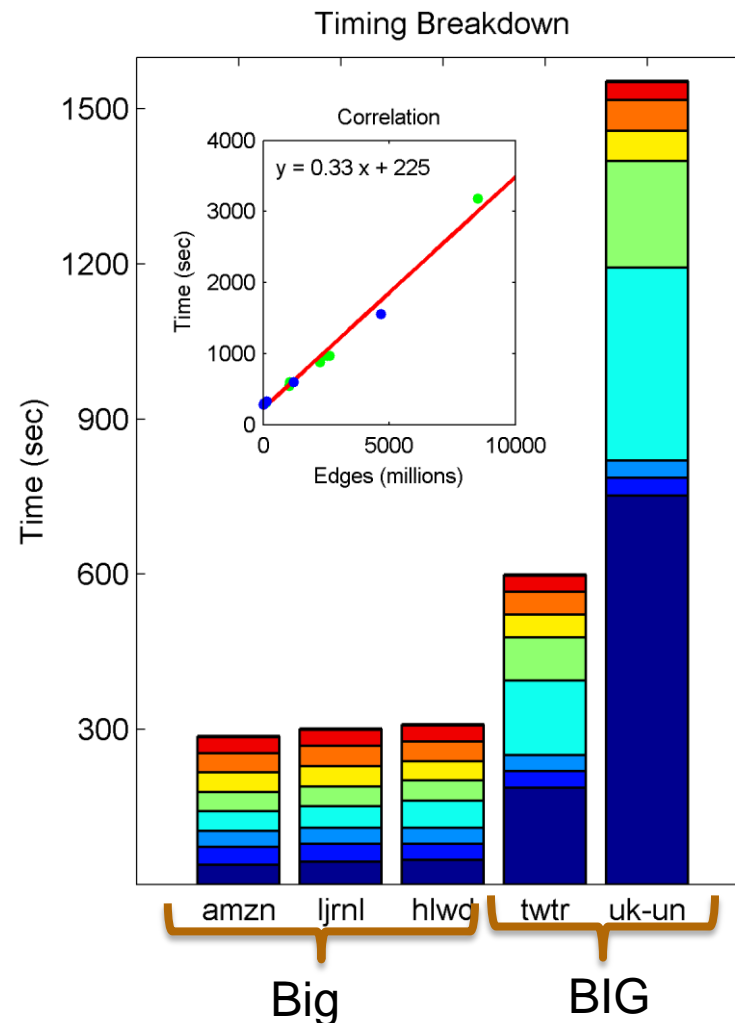
Large Data Sets and Machines

- 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



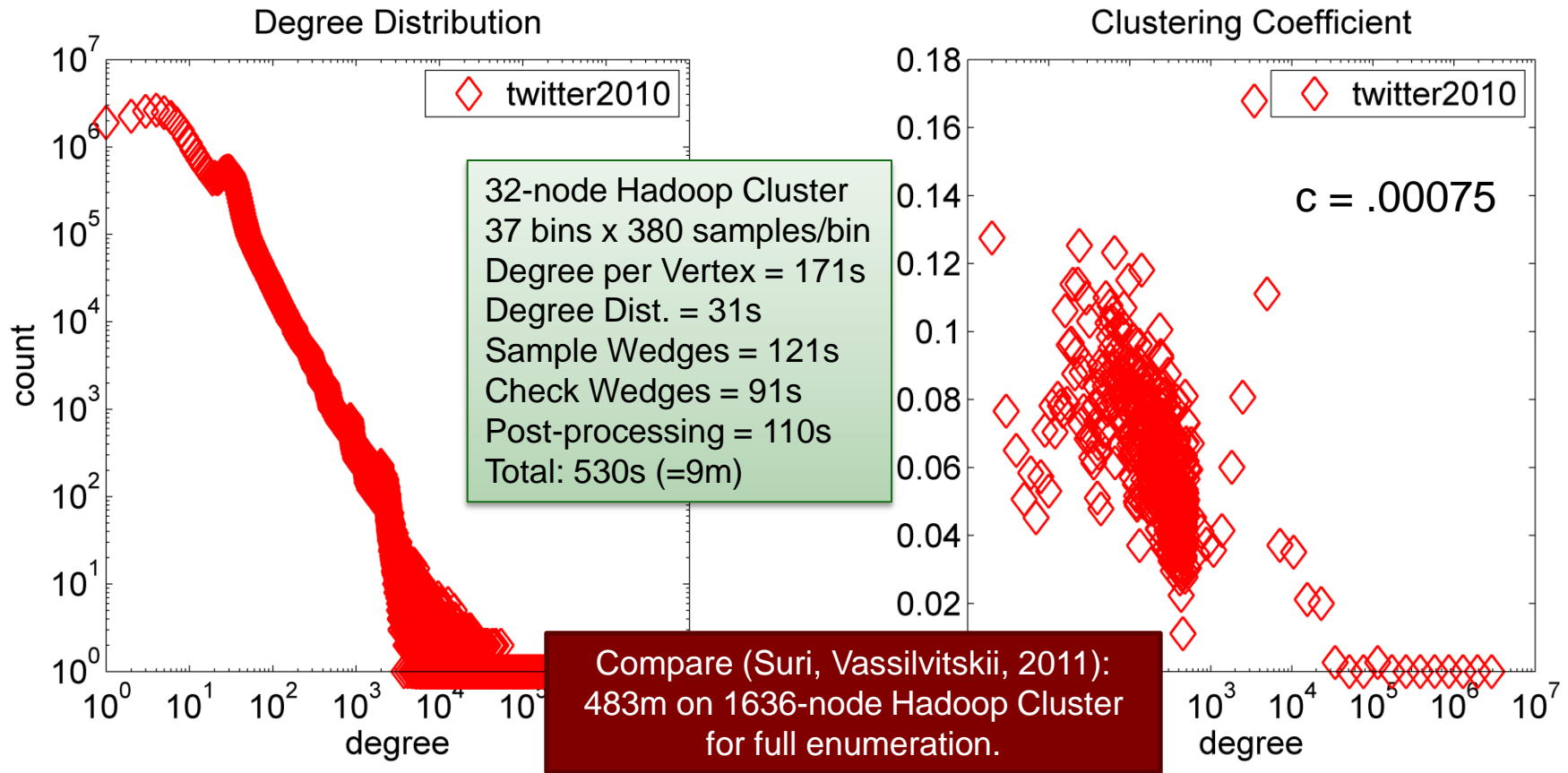
Wedge Sampling for BIG Graphs

- 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 Follower Network

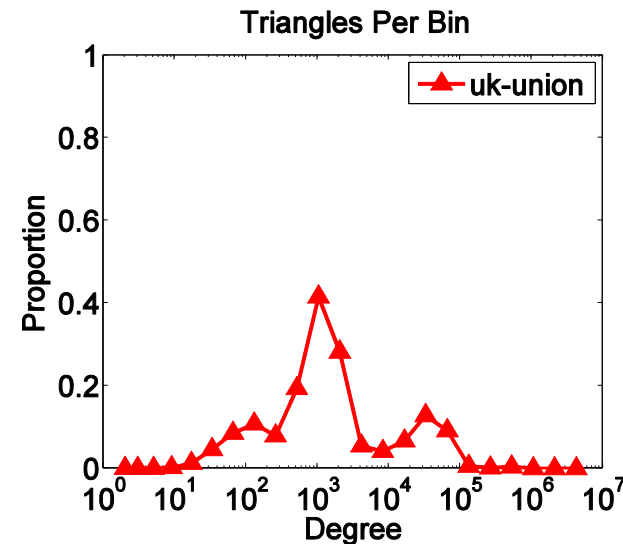
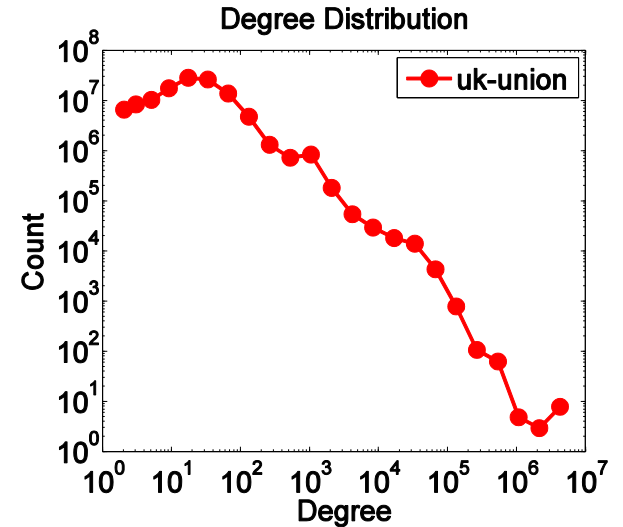
twitter-2010: 41M nodes, 1.2B undirected edges, 123T wedges, 31B triangles
Nodes are users. Connected if one follows the other.



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 of 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.



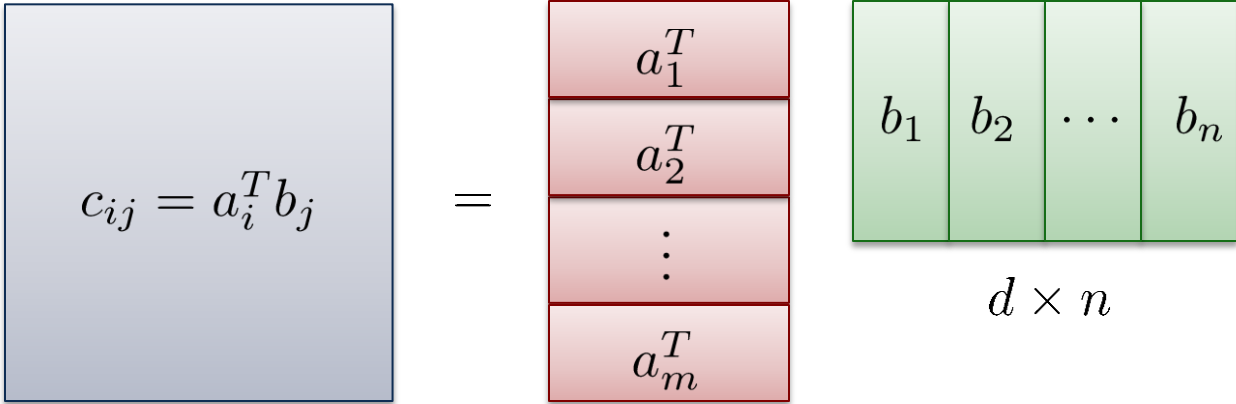
Problem: Maximum All-pairs Dot-product (MAD) Search

MAD Search

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

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

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

$$C = A^T B$$


$c_{ij} = a_i^T b_j$

$m \times n$

a_1^T

a_2^T

\vdots

a_m^T

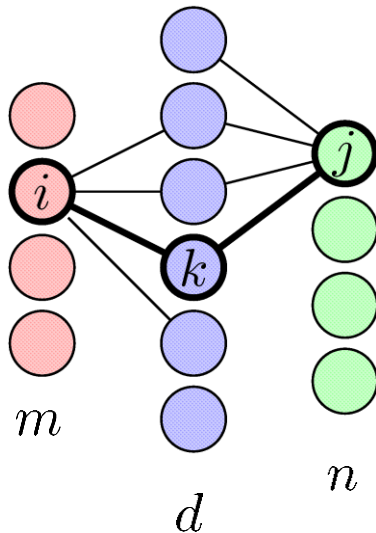
$m \times d$

b_1 b_2 \dots b_n

$d \times n$

MAD Search: Tripartite Graph Representation for Binary Inputs

Tripartite Graph



Showing only edges corresponding to vector a_i in A and vector b_j in B .

Binary Inputs

$$A = [a_1 \cdots a_m] \text{ and } B = [b_1 \cdots b_n]$$

$$C = A^T B$$

$c_{ij} = \#$ wedges between nodes i and j :

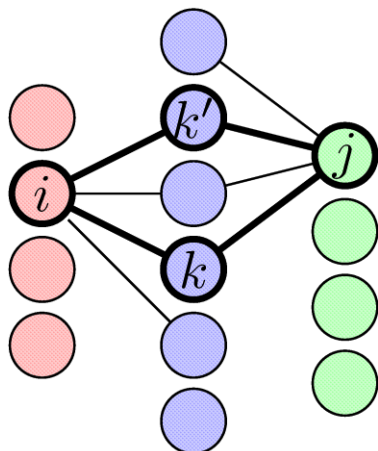
$$c_{ij} = a_i^T b_j = |\{k \mid k \in \mathcal{N}_i^A \cap \mathcal{N}_j^B\}|$$

If there is a “wedge” (i, k, j) , then $c_{ij} \geq 1$

$$\text{Prob}(\text{wedge } (i, *, j)) \propto c_{ij}$$

Pair of Wedges = Diamond

Diamond = Two Intersecting Wedges



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

$c_{ij}^2 = \#$ diamonds between nodes i and j

$\text{Prob}(\text{diamond } (*, i, *, j)) \propto c_{ij}^2$

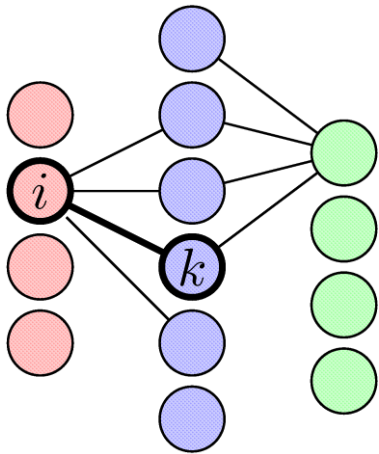
Diamond: (k', i, k, j)

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.

Sampling Random Diamonds for Binary Inputs

Enumerating all diamonds is too expensive. Instead, we need an inexpensive way to sample.

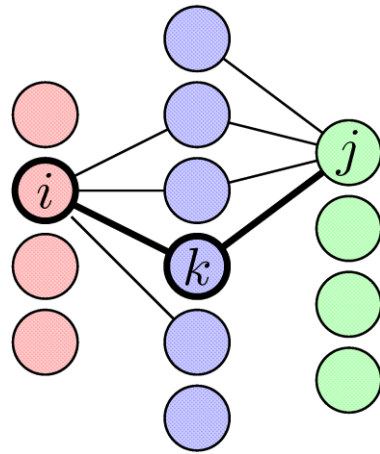
Step 1: Choose edge $(i,k) \propto w_{ik}$



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

$$w_{ki} = \deg_i^A \deg_k^B$$

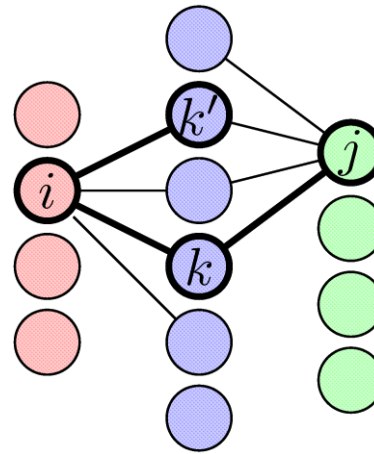
Step 2: Choose neighbor of k



Sample j from \mathcal{N}_k^B

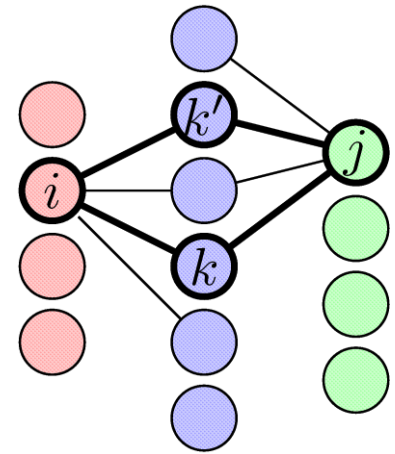
Three-path sampling: Jha et al. WWW 2015

Step 3: Choose neighbor of i



Sample k' from \mathcal{N}_i^A

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

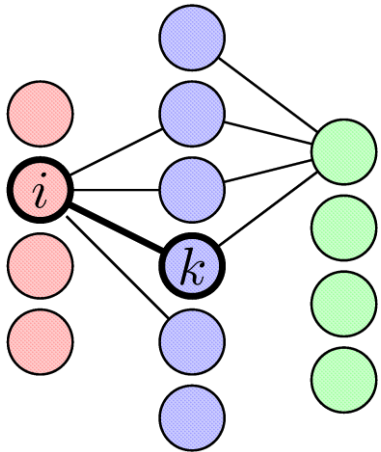


If edges exists, diamond complete:

$$x_{ij} \leftarrow x_{ij} + 1$$

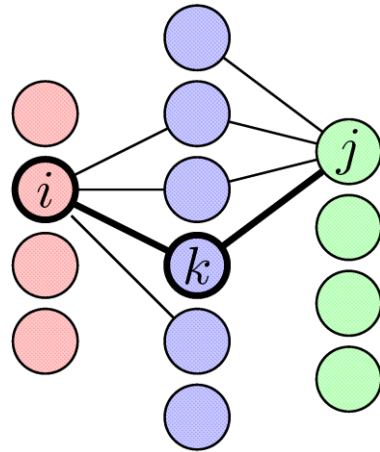
Sampling Random Diamonds for General Inputs

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



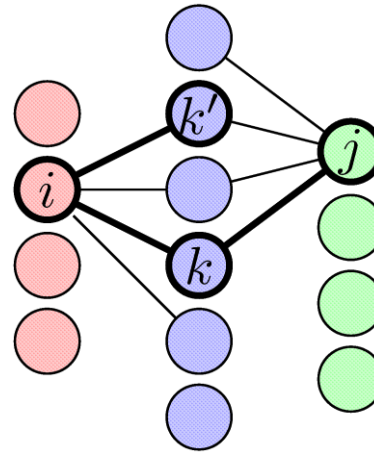
$$w_{ki} = \frac{|a_{ki}|}{\|a_{*i}\|_1} \|b_{k*}\|_1$$

Step 2: Choose neighbor of k



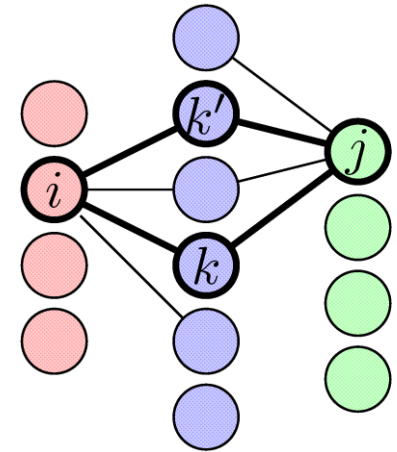
Sample j with probability $\frac{|b_{kj}|}{\|b_{k*}\|_1}$

Step 3: Choose neighbor of i



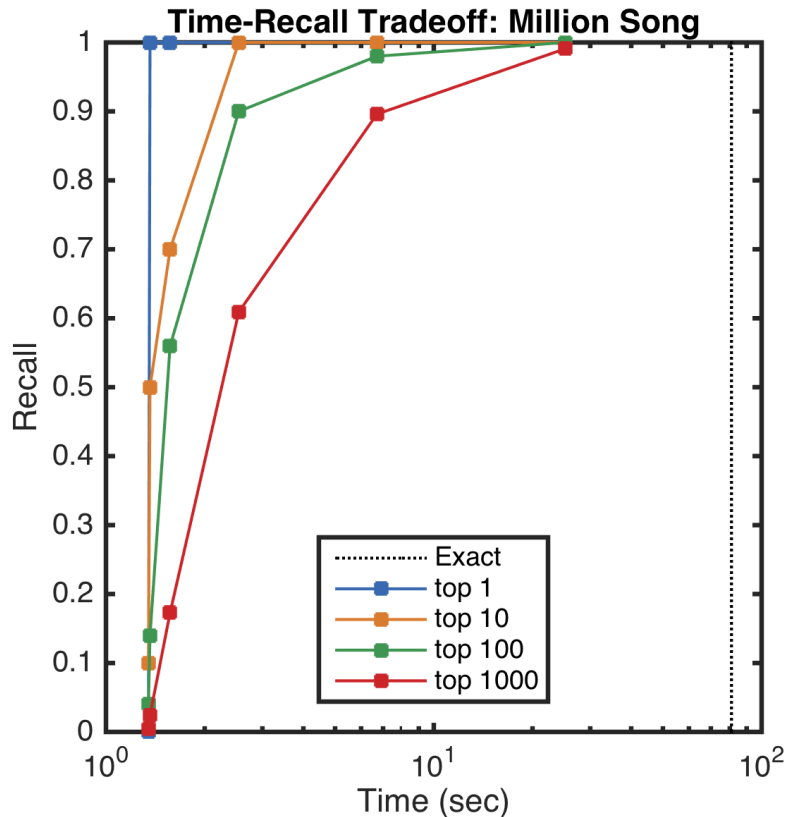
Sample k' with probability $\frac{|a_{k'i}|}{\|a_{*i}\|_1}$

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



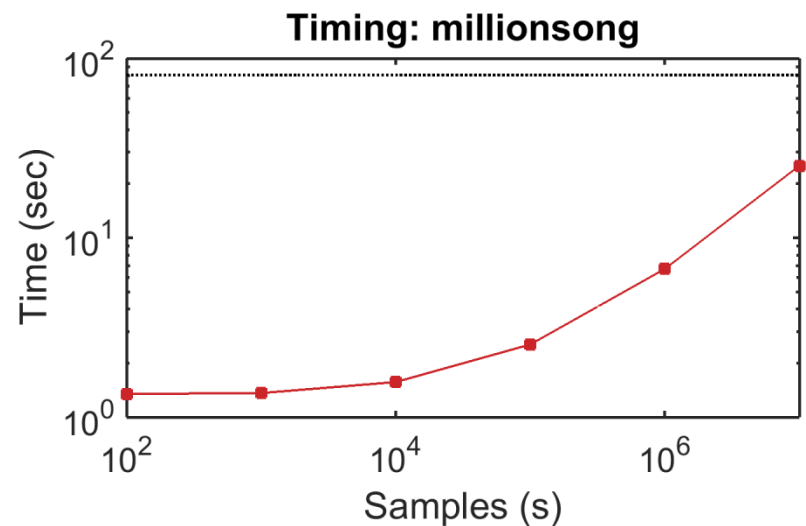
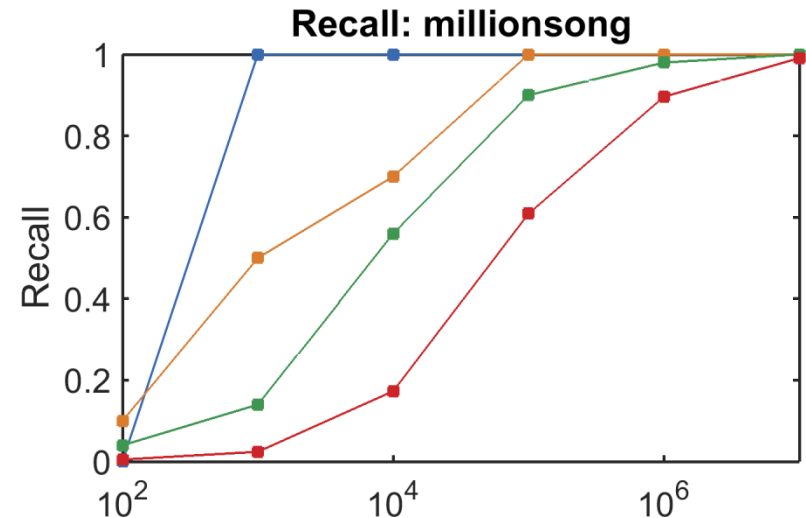
$A = B =$ user-song listens
 $d = 1\text{M}$ users, $m = n = 350\text{K}$ songs
 $\text{nnz}(A) = 48\text{M}$
 $C = A^T A =$ song similarity
max samples = 10^7

- 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 $j = 1, \dots, n$
 - Exploit symmetry for $B = A$
- CSPARSE from Tim Davis
 - Modified cs_multiply function

Results: Million Song

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

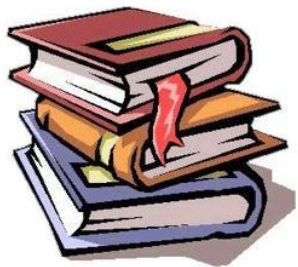


Motivating Example: Free Samples!

Book seller wants to increase reviews on its site.



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



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

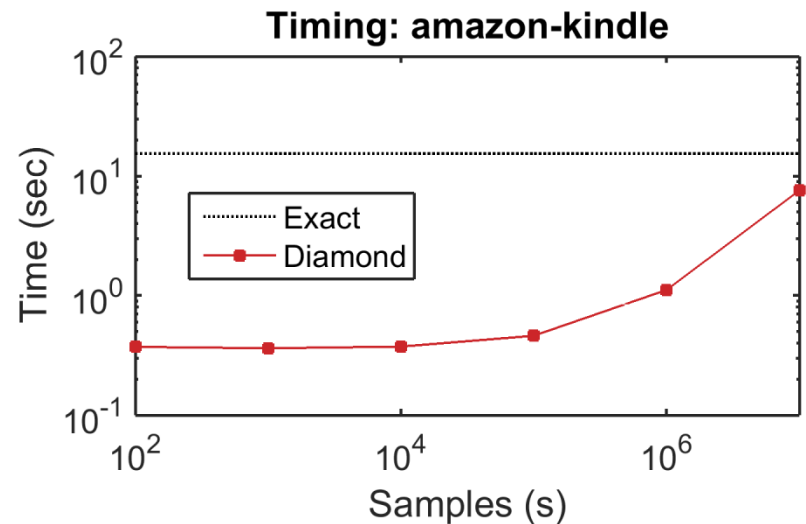
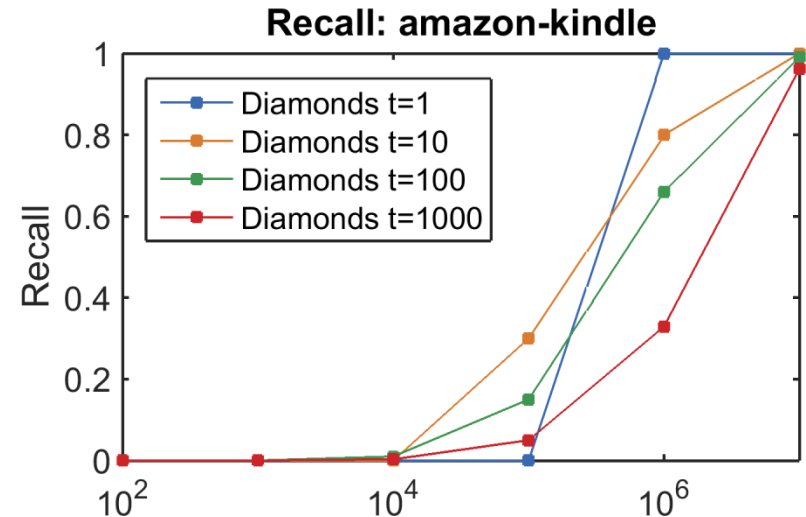
Send free books to users who have written reviews of similar books.



$c_{ij} = \#$ books reviewed by person i
that are similar to book j

Results: Amazon Kindle

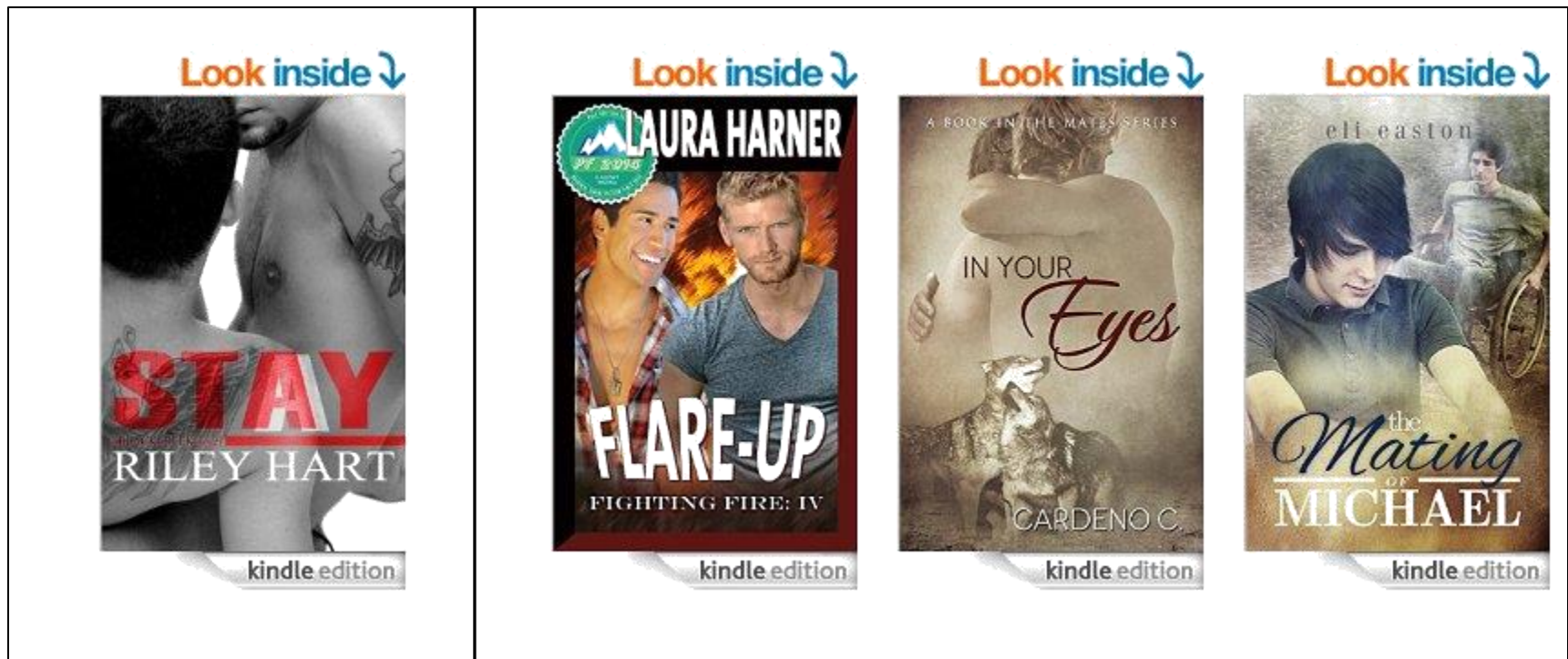
- Product recommendations
- A is sparse reviewer-by-book rating matrix and B is sparse book similarity matrix
 - $m = 1,406,916$ reviewers
 - $n = d = 430,532$ books
 - $\text{nnz}(A) = 3,205,546$
 - $\text{nnz}(B) = 11,012,558$
- Max entry in C : 3.2×10^3
- # Wedges = 1.2×10^{12}
- # Samples = $O(10^{12})$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



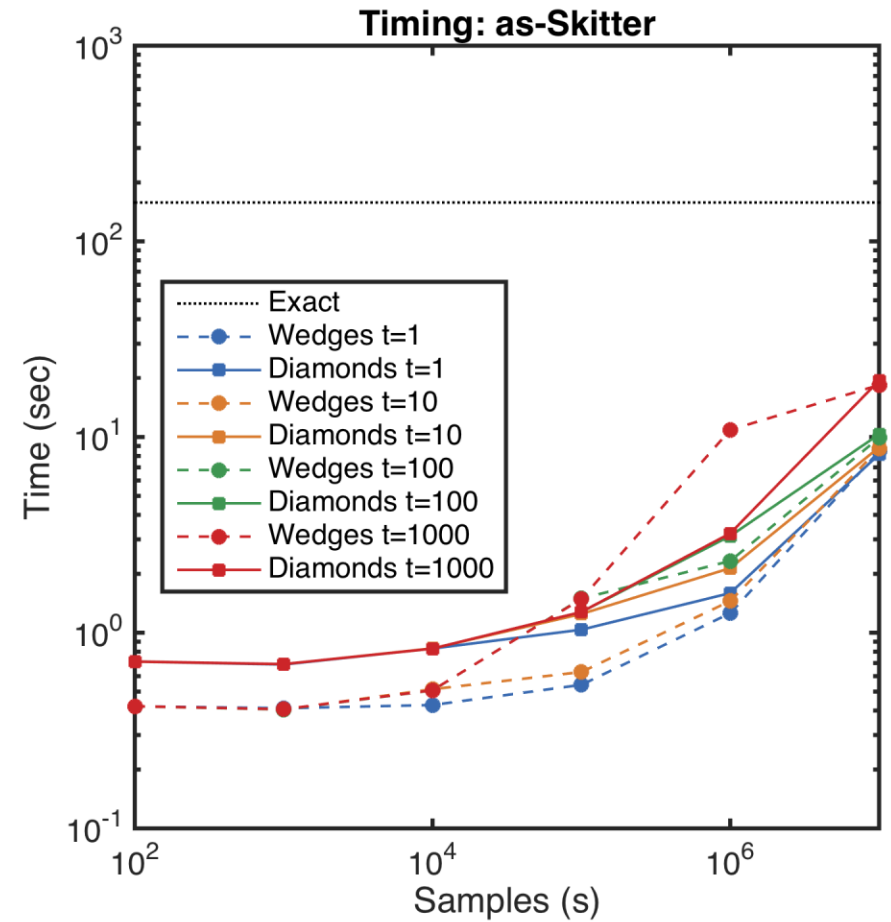
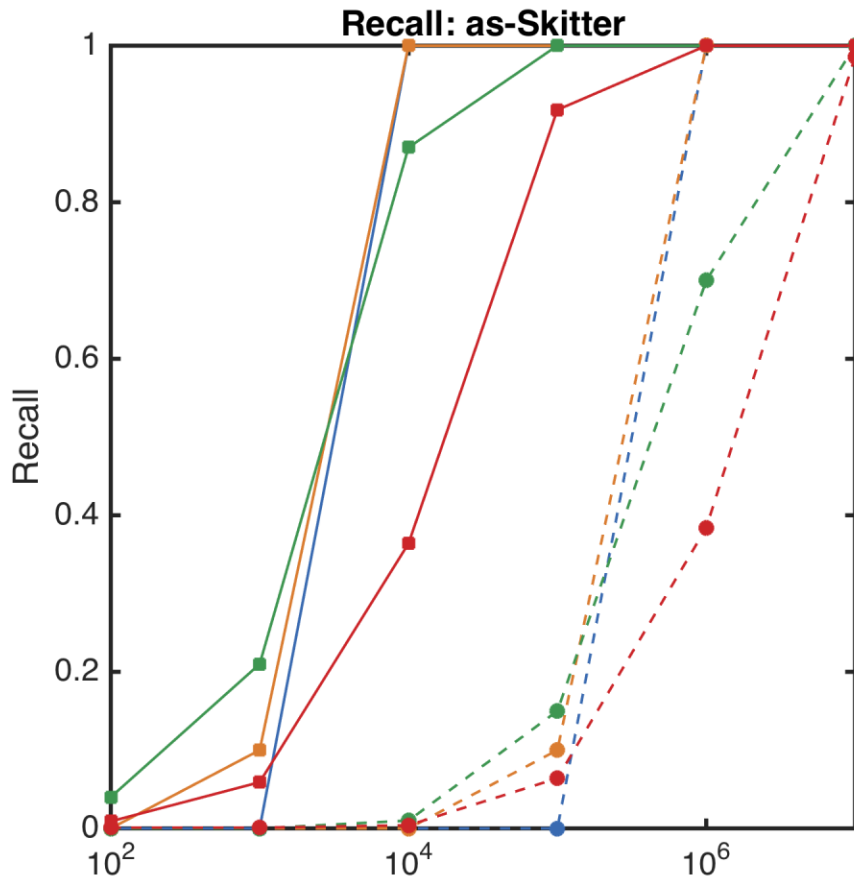
Example Reviewer-Book Recommendation

Recommended Book

Recommended User ID = 1317513
Previous Reviews Include...



Comparison to Wedge Sampling



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?
- Tamara Kolda
tgkolda@sandia.gov
- **Best Paper Prize!** C. Seshadhri, A. Pinar and T. G. Kolda, *Triadic Measures on Graphs: The Power of Wedge Sampling*, SDM'13, Austin, TX, pp. 10-18, 2013, [doi:10.1137/1.9781611972832.2](https://doi.org/10.1137/1.9781611972832.2)
 - C. Seshadhri, A. Pinar and T. G. Kolda, *Wedge Sampling for Computing Clustering Coefficients and Triangle Counts on Large Graphs*, Statistical Analysis and Data Mining 7(4):294-307, 2014, [doi:10.1002/sam.11224](https://doi.org/10.1002/sam.11224)
 - T. G. Kolda, A. Pinar, T. Plantenga, C. Seshadhri and C. Task, *Counting Triangles in Massive Graphs with MapReduce*, SIAM J. Scientific Computing 36(5):S44-S77, 2014, [doi:10.1137/13090729X](https://doi.org/10.1137/13090729X)
 - **Best Paper Prize!** G. Ballard, A. Pinar, T. G. Kolda and C. Seshadhri, *Diamond Sampling for Approximate Maximum All-pairs Dot-product (MAD) Search*, ICDM 2015, Atlantic City, NJ, 2015, <http://arxiv.org/abs/1506.03872>
 - **Tensors for Incomplete Data:** E. Acar, D. M. Dunlavy, T. G. Kolda and M. Mørup, *Scalable Tensor Factorizations for Incomplete Data*, Chemometrics and Intelligent Laboratory Systems, 106(1):41-56, 2011, [doi:10.1016/j.chemolab.2010.08.004](https://doi.org/10.1016/j.chemolab.2010.08.004)