SAND2013-9066C

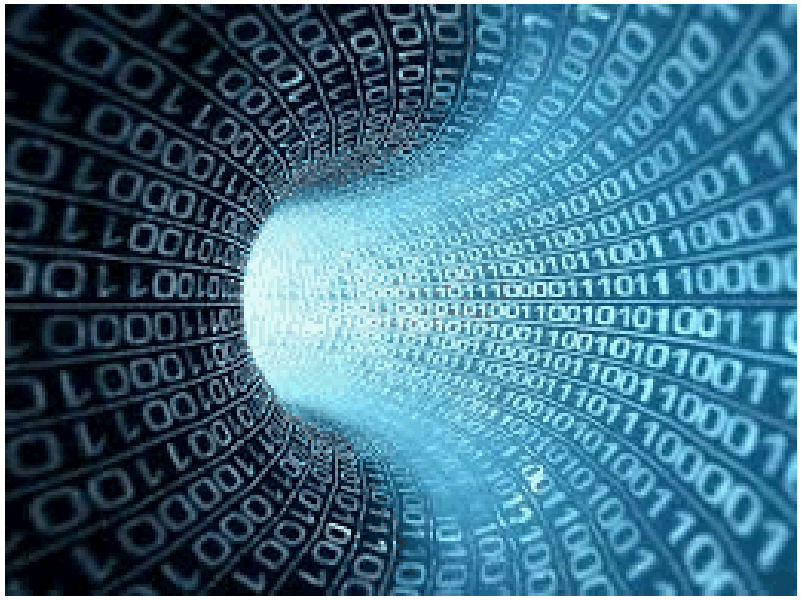
Sampling Algorithms to Count Frequent Patterns in Graphs

Ali Pinar

Sandia National Laboratories

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sampling may be the key to process large data sets

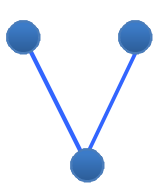


Source: http://www.greenbookblog.org/wp-content/uploads/2012/03/big-data.jpg

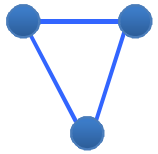
- Sizes of the modern data sets redefine the landscape for algorithmic research.
  - Single pass through the data may be a luxury.
  - There may not be sufficient memory to store all the data.
- In many applications the speed of data is the challenge.
- The goal of sublinear algorithms is to provide
  - good estimations with error/confidence bounds,
  - by looking at a small portion of the data.
- The trick is sampling…
  - intelligent sampling.

Triangles are critical for graph analysis

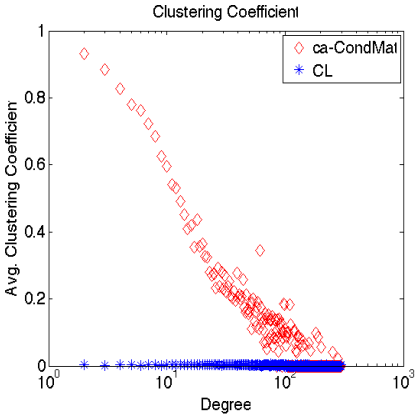
- Interpreted in many different ways in social sciences.
  - Identifier for bridges between communities.
  - Likelihood to go against norms
- Applied to spam detection
- Used to compare graphs
- Proposed as a guide for community structure.
- Stated as a core feature for graph models [Vivar&Banks11]
  - Cornerstone for Block Two-level Erdos-Renyi (BTER)
- Rich set of algorithmic results
  - Algorithms, runtime analysis, streaming algorithms, MapReduce, …

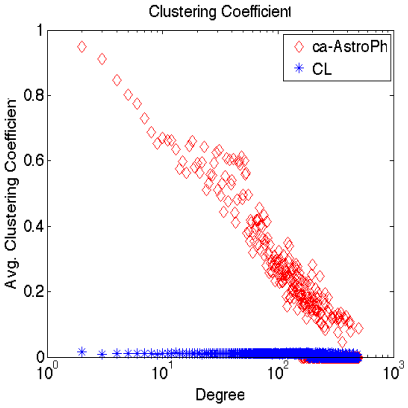


Open wedge

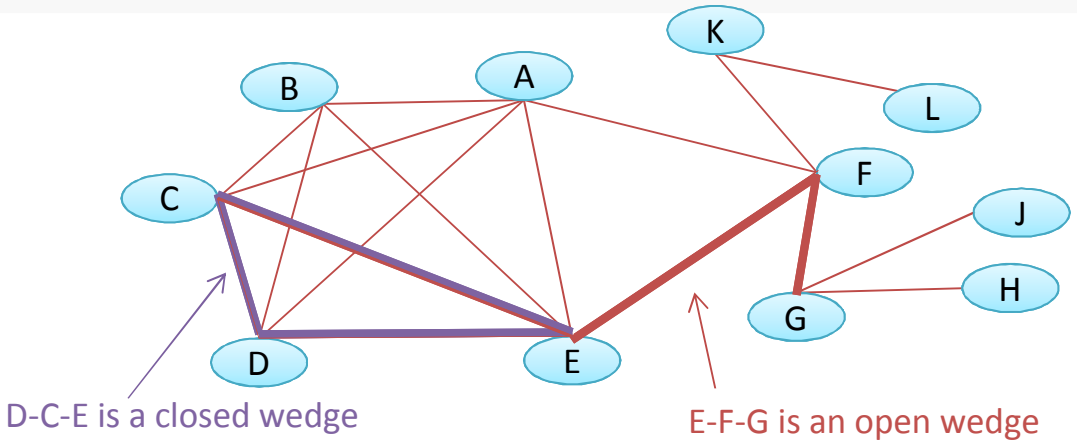


Closed wedge, (i.e., triangle)





Algorithms for important metrics: Clustering coefficients for large graphs



$$c = \frac{3 \times \#\text{ triangles in graph}}{\#\text{ wedges in graph}}$$

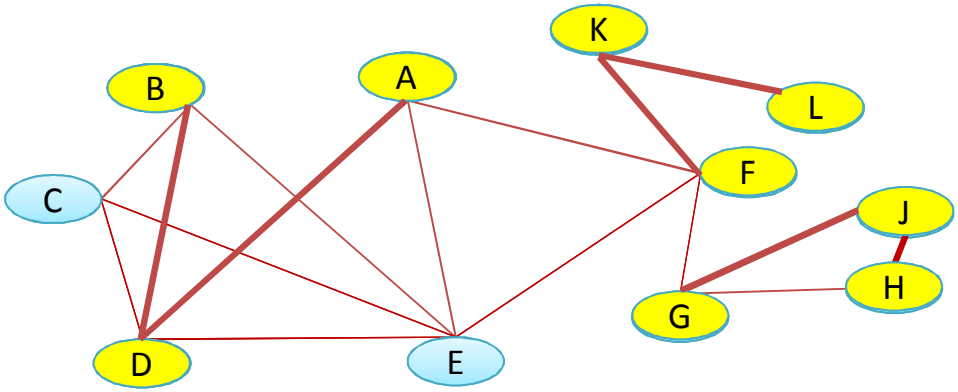
i.e., fraction of wedges that are closed

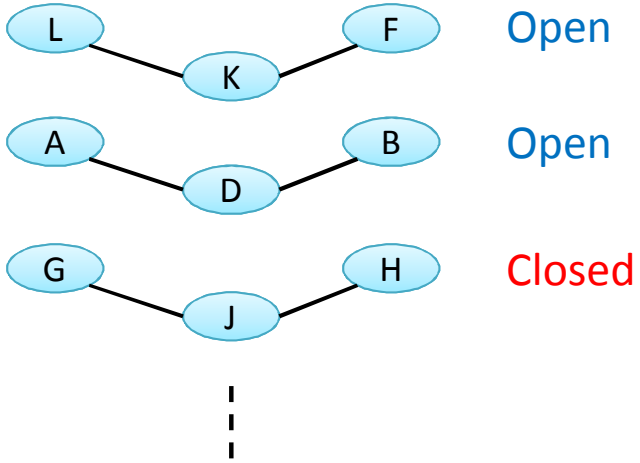
Enumeration: Find *every* wedge. Check if each is closed.
c = # closed wedges / # wedges

Sampling: Sample a few wedges (uniformly). Check if each is closed.
c = # closed sampled wedges / # sampled wedges

Seshadhri, Pinar, Kolda, *SIAM Intl. Conf. Data Mining* 2013, Best Research Paper award

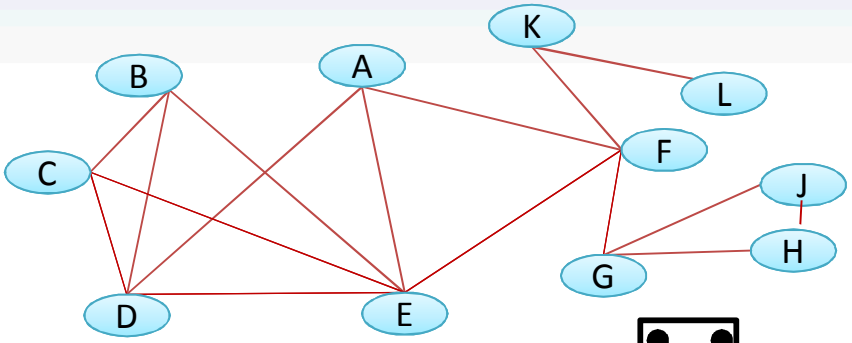
Wedge sampling for τ

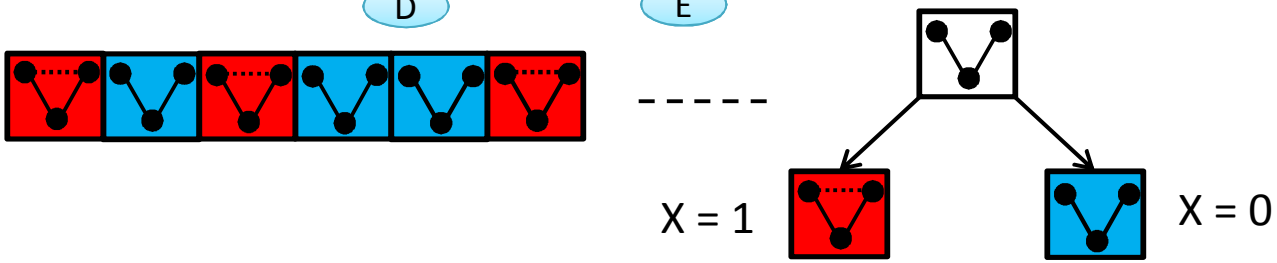




- τ = 3T/W = fraction of closed wedges
- Consider list of all wedges, indexed with open/closed

Wedge sampling for τ

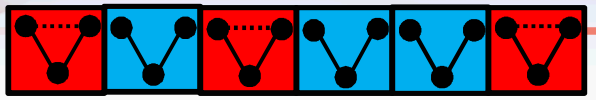


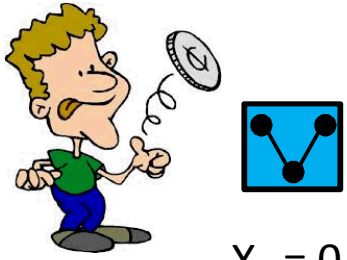


- τ = 3T/W = fraction of closed wedges
- Consider list of all wedges, indexed with open/closed
- Pick a uniform random wedge. X = 1 if wedge is closed. Else X = 0
- X is Bernoulli random variable

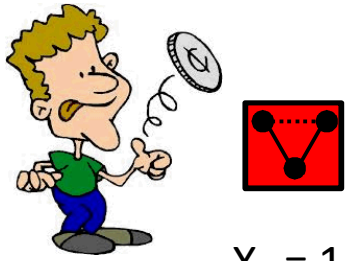
  and E[X] = fraction of closed wedges = τ = 3T/W

Repeat, repeat, repeat

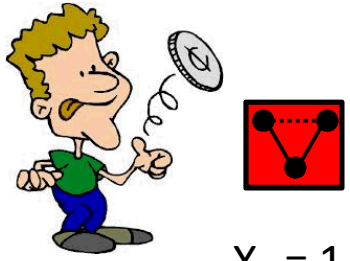




$X_1 = 0$



$X_1 = 1$

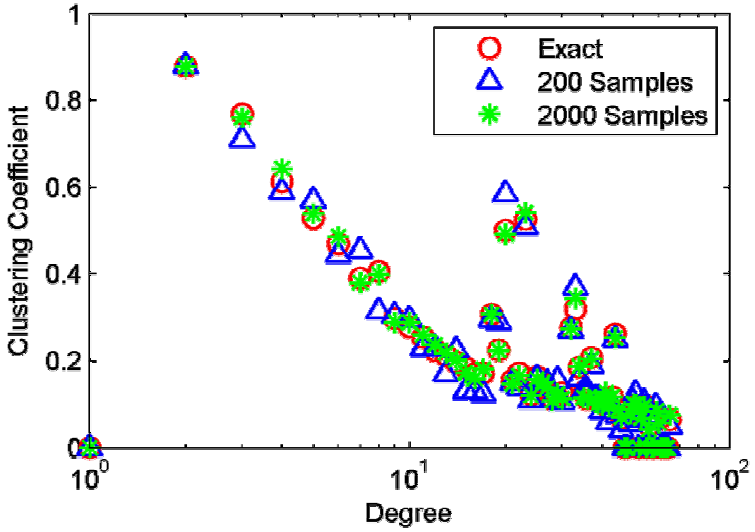


$X_2 = 1$

- Perform k independent experiments. Let $Y = (1/k) \sum_i X_i$
  - Y is fraction of closed wedges in sample
  - $E[Y] = \tau$. Y converges to τ as k grows
- [Chernoff-Hoeffding]: $\Pr[|Y - \tau| > \varepsilon] < e^{-k\varepsilon^2}$
  - $k = \varepsilon^{-2} \log (1/\delta)$. With prob > 1- δ, estimate is accurate within ε
  - With 38K samples, error < 0.01 with prob > 0.999
  - Number of samples independent of graph size

Benefits of Wedge Sampling

- Bounded error for specified sample size and desired confidence
- Work is O(# edges) vs O(# wedges)
- **1000X average speedup** versus enumeration, k = 32,000 (ε = 0.011)
- Faster than edge sampling (Doulion) with less variance
- Can also compute clustering coefficient per degree



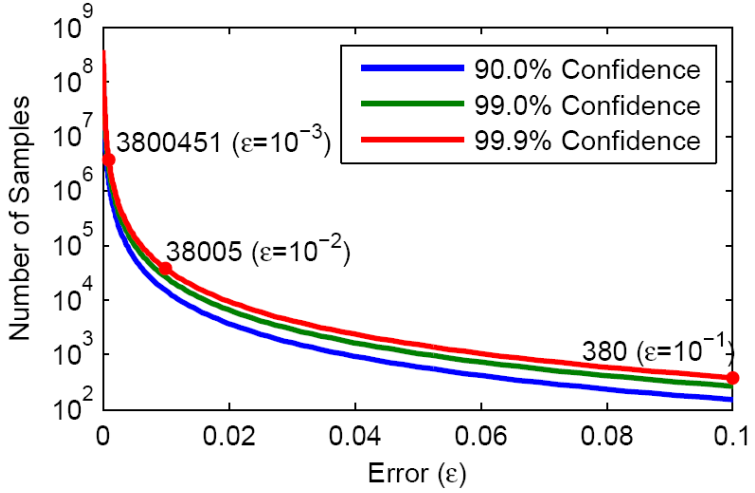
Bounded Error: Hoeffding's Inequality

Theorem: (Hoeffding 1963) Let $X_1, X_2, \ldots, X_k$ in [0,1] be independent random variables. Define the sample mean: $\bar{X} = \frac{1}{k}\sum_{i=1}^{k} X_i$

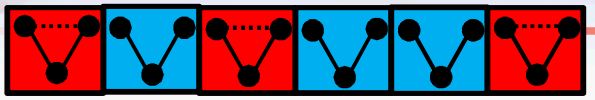
Let μ be the true mean. Then for ε in (0,1),

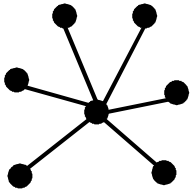
$$\text{Prob}\left\{|\bar{X} - \mu| \geq \epsilon\right\} \leq \delta \equiv 2\exp(-2k\epsilon^2)$$

Hence, for a given error ε and confidence 1-δ, we just need to set $k = \lceil 0.5\epsilon^{-2}\log(2/\delta)\rceil$

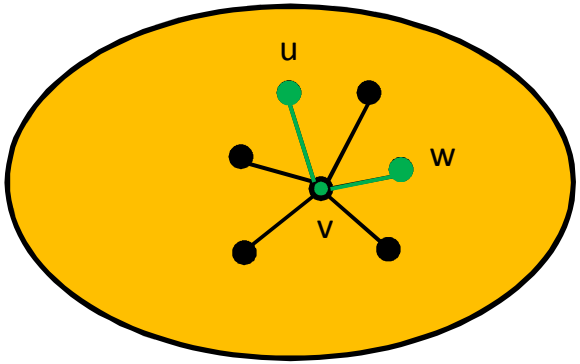


Don’t generate list



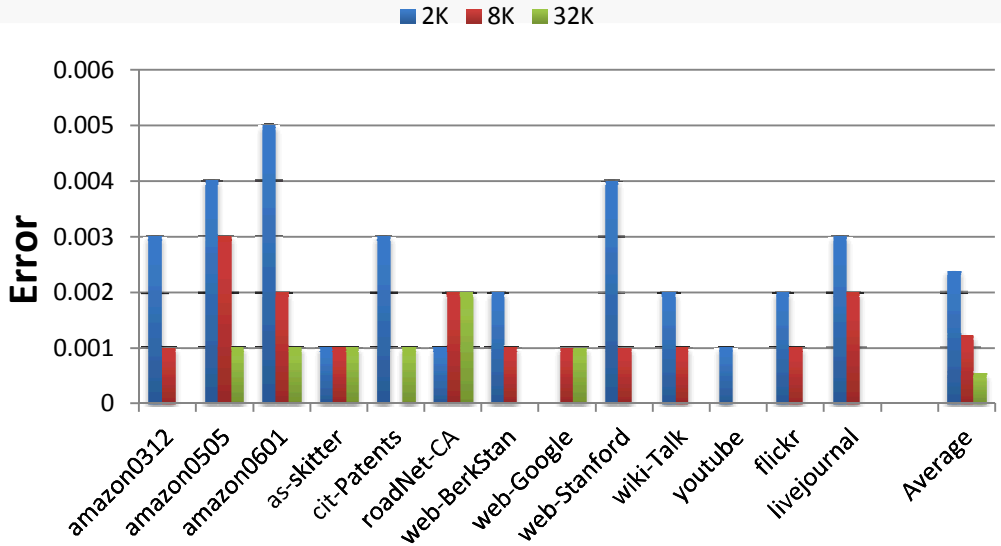


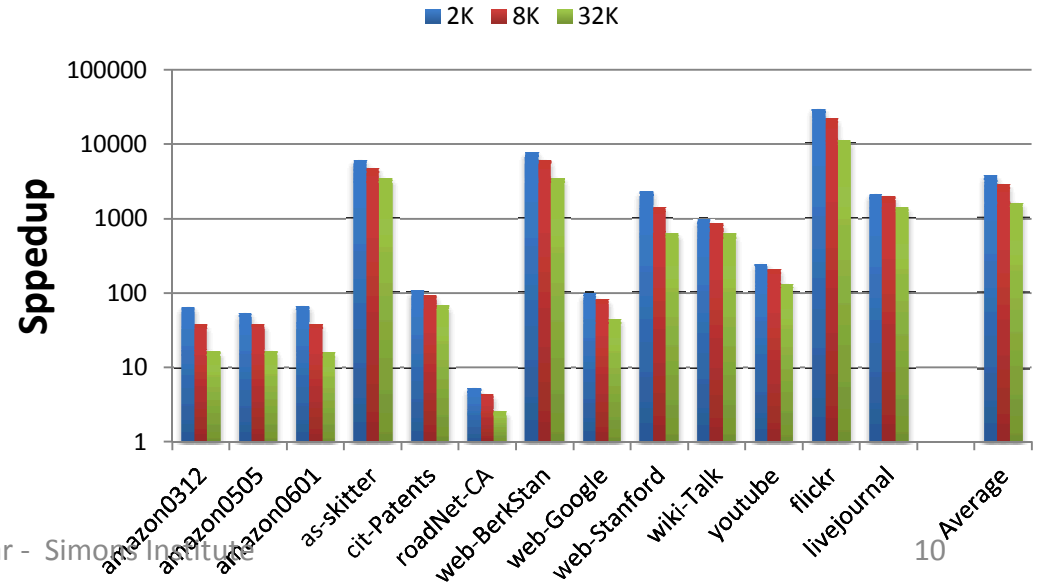
$$W_v = \binom{d_v}{2}$$



- But list of wedges not possible to generate. So how to get random wedge?
- Pick vertex v with probability W_v/W
- Pick two uniform random neighbors of v to get wedge (u,v,w)
  - This is a uniform random wedge
- So simply repeat this many times to get a set of wedges. Output fraction of closed wedges as estimate for τ

Wedge sampling is effective in practice

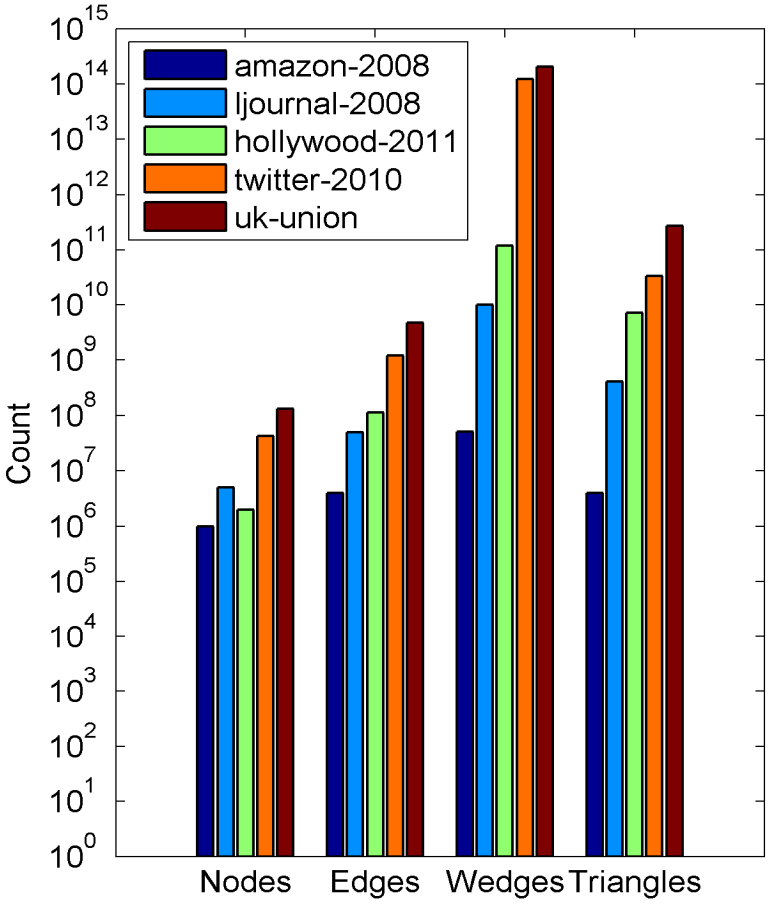




Data Sets and Machines

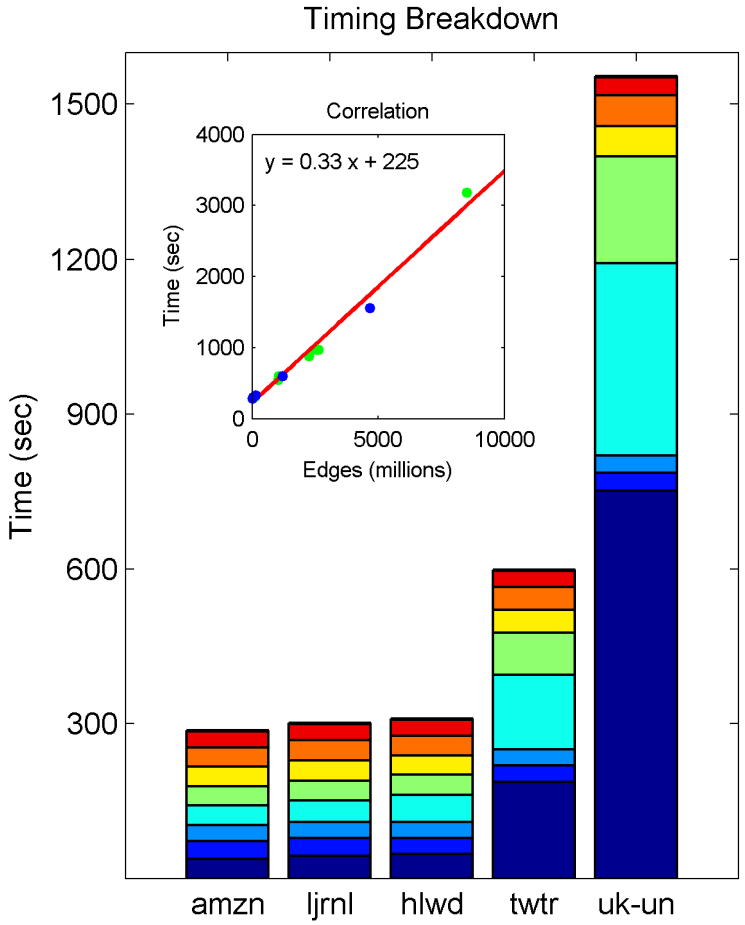
- 5 real-world networks
  - Source: Laboratory for Web Algorithms (Italy)
  - Largest: 132M nodes, 4.6B edges
  - Observe: # wedges À # edges!

- 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

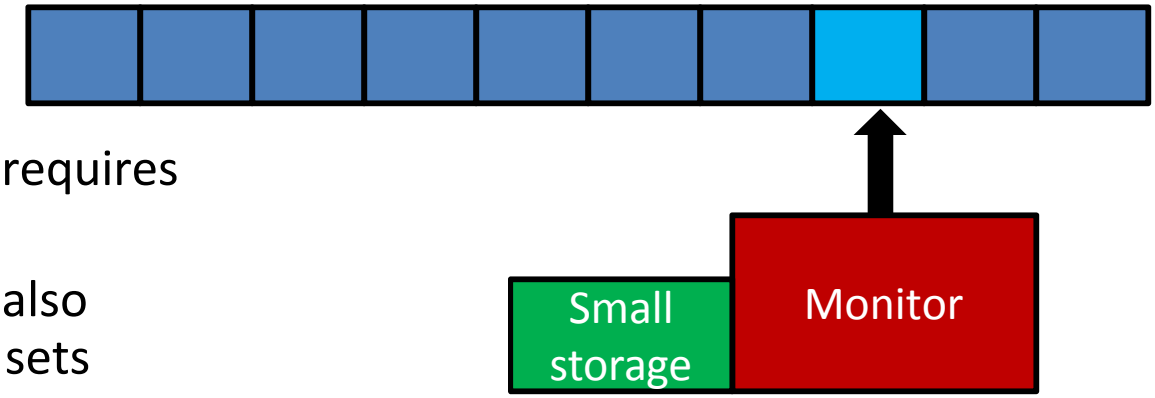
- 32-node Hadoop cluster results using wedge sampling
  - Logarithmic bins
  - 2000 samples per bin
- "BIG" graphs benefit from Hadoop
  - Merely "Big" graphs don't require Hadoop – just shown as examples
- Compare twitter times
  - Sampling: 10 mins on 32-node Hadoop cluster
  - Enumeration: 483 mins 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



Kolda, Pinar, Plantenga, Seshadhri, Task, arXiv:1301.5886, 2013

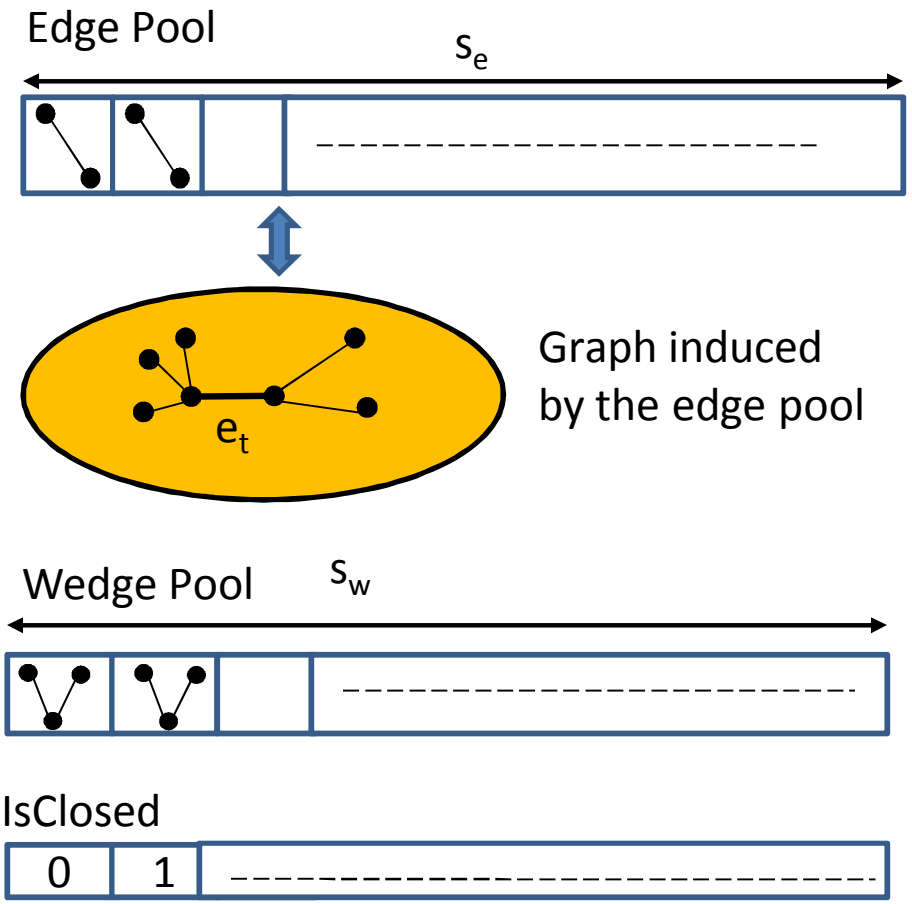
What if we observe the data as a stream of edges?

- Many data analysis problems deal with data streams.
  - Situational awareness requires real time analysis.
- Streaming algorithms are also used to analyze large data sets with limited memory.
  - Multiple passes may be feasible.
- Algorithmically
  - We see each data point only once.
  - We either take action, or forever hold our peace.
- Not all problems are amenable to streaming analysis.
  - We cannot find needle in a haystack
  - But we can count frequent items, such as triangles



Wedge sampling in a streaming world

- Challenges and solutions:
  - Challenge: How do we generate a random sample of wedges?
  - Solution: Keep a random edge pool and trust in the birthday paradox.
  - Challenge: How do we keep a random pool from a stream?
  - Solution: use reservoir sampling.
  - Challenge: What if the closing edge of a wedge has already gone by?
  - Solution: adjust the clustering coefficient by multiplying by 3.



Jha, Seshadhri, Pinar, *KDD* 2013, Best Student Paper award

How does the algorithm work?

For the t-th edge e_t

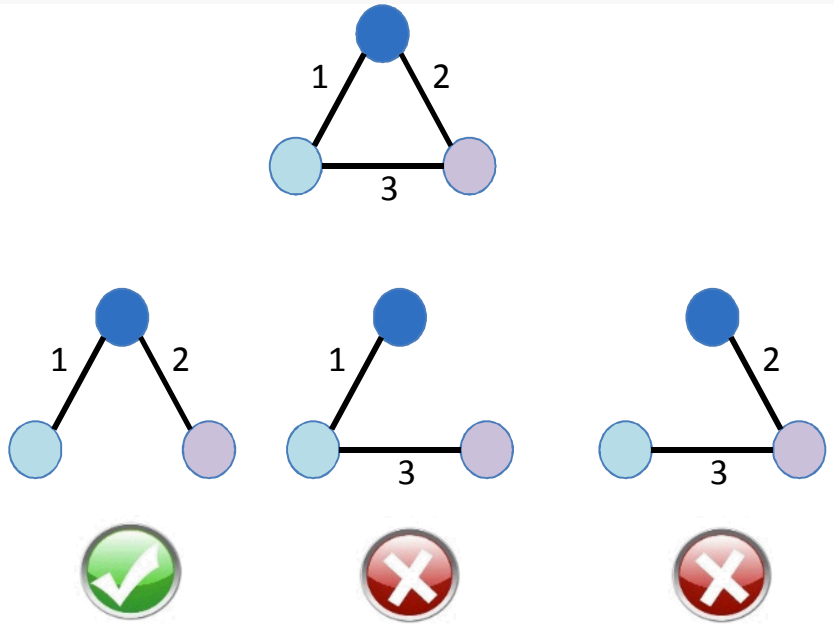
1. Mark wedges closed this wedge as closed.
2. Reservoir sampling: maintain a random sample of edges seen so far.
   - Flip coin with head prob = $1 - (1-1/t)^{se}$. If tails, continue to next edge, if not pick an edge to replace.
3. Update the estimate for the number of wedges
4. Update the wedge pool with the new wedges formed and the new estimate for the #wedges.

At any point in the algorithm

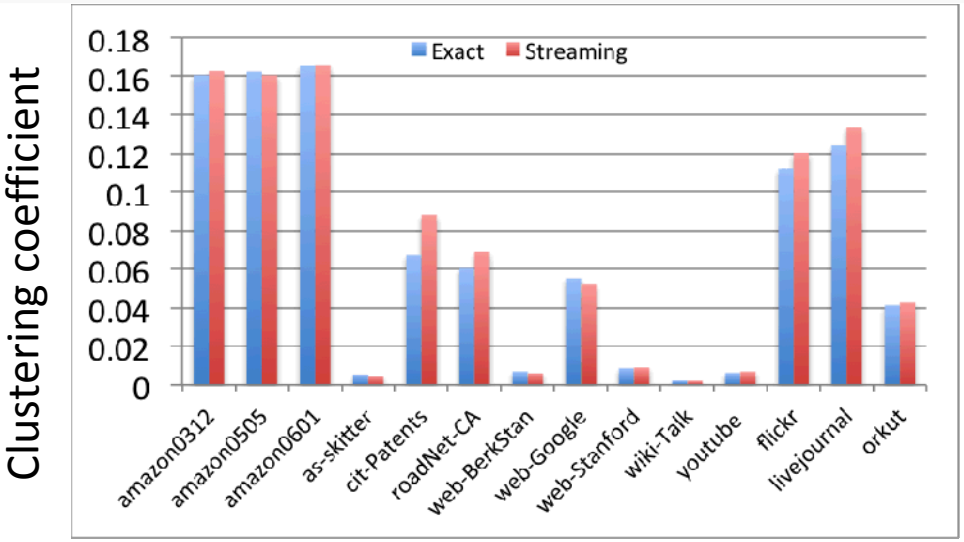
- The clustering coefficient is estimated as 3* fraction of closed wedges
- The number of wedges is estimated using the birthday paradox and the wedges defined by edges in the edge pool.
- The number of triangles is estimated as the product of estimates of the wedge count and the clustering coefficient.

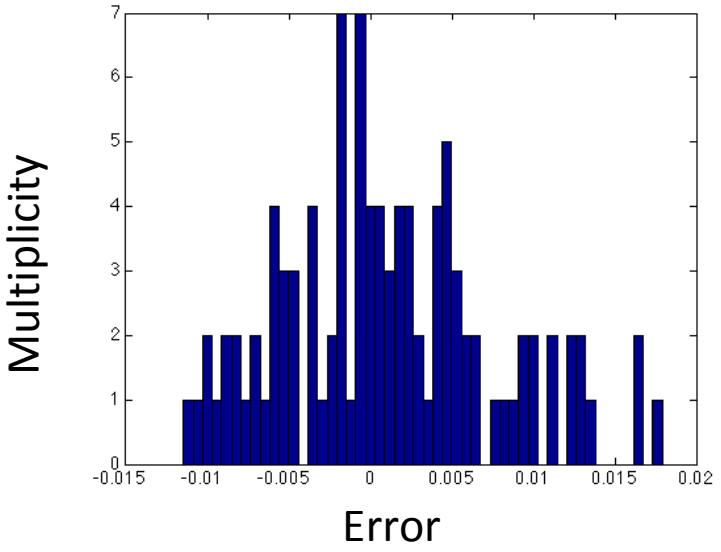
Making up for wedges closed by earlier edges

- Each triangle comprises of 3 wedges.
- In the original wedge sampling, we were able to detect any wedge as closed.
- In the streaming algorithm, we can only detect 1 of the 3 as closed.
- Since wedges are selected randomly, the expected closure rate is 3* the closure rate of the wedge pool.



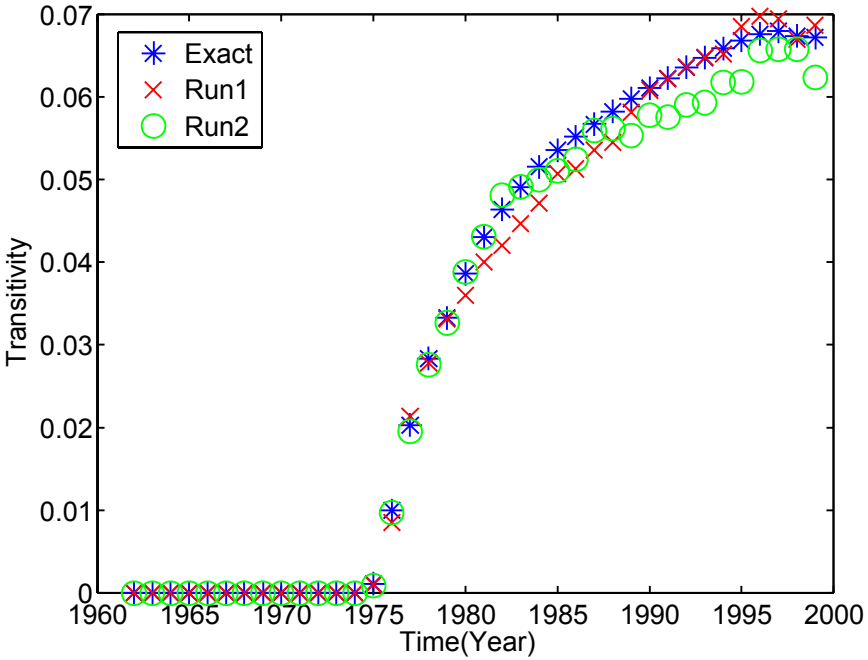
Streaming algorithm is effective in practice

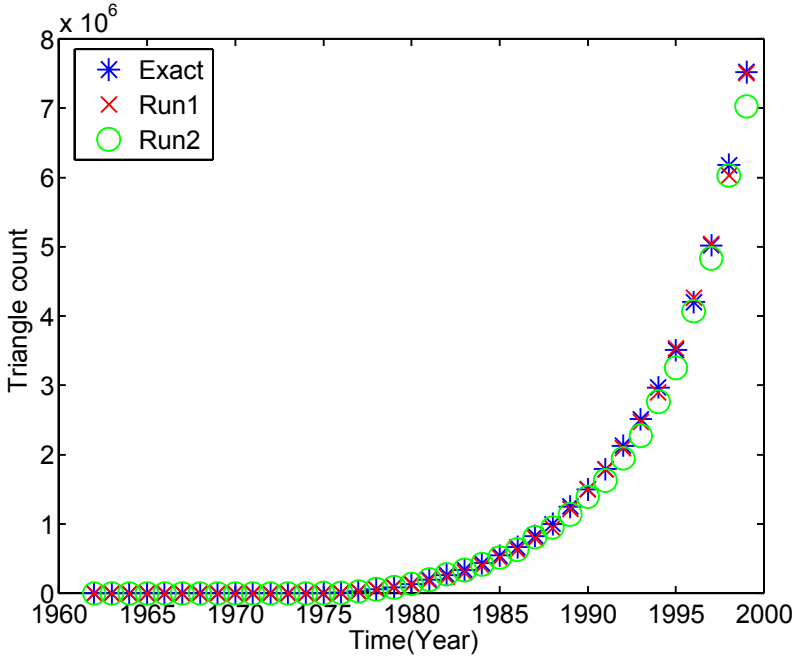




- Experiments on public data sets
- Edge pool size: 20K; Wedges pool size is: 20K
  - Pool sizes are independent of the graph size.
- The estimates are accurate.
- The variance is small.

Streaming algorithm provides a running estimate

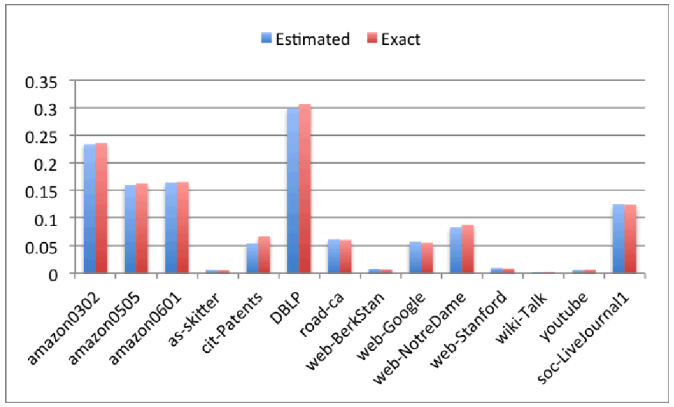




- Results on the patent citation network
  - 3.8M vertices, 16.5M edges.
- The algorithm provides accurate running estimates.

Next step: Streaming multigraphs

- Many graphs are a stream of *repeated* edges. (Emails, data transfers, co-authorship etc.)
- Generalized our algorithm for multigraphs.
  - Used random hashing to detect multiple instances.
  - Devised un unbiasing technique to avoid stream order sensitivity.
    - aaabbbccc vs. abcabcabc
- Processed the DBLP raw data
  - 1.2M vertices, 5.1M simple edges, 9.0 repeated edges,11.4M triangles transitivity= 0.1743.
  - Estimate with 30K edges and 30K wedges 11.3 triangles transitivity= 0.1733
- Also effective in artificially generated sets



Conclusions and Future Work

- If the data get bigger, the algorithms should get smarter.
  - Data will grow faster than computational resources.
  - By 2020, there will be 5.2TB/person.
- Sampling will be instrumental in large scale data analysis.
  - Error/confidence bounds are critical.
- We have extended our sampling ideas to handle streaming data.
- We want to extend the work for
  - Graphs with repeated edges
  - Analyzing higher patterns

References

The framework

- Parameters to streaming algorithm
  - Size of the edge pool
  - Size of the wedge pool
- For each edge e_t in stream
  - Update storage
  - Let fraction of true isClosed entries be p
  - The clustering coefficient is estimated as 3p.
  - The number of wedges is estimated using the birthday paradox and the wedges defined by edges in the edge pool.
  - The number of triangles is estimated as the product of estimates of the wedge count and the clustering coefficient.

- BACK –UP SLIDES

The algorithm: updating on e_t

1. For every wedge_res[i] closed by e_t, set isClosed[i] = TRUE
2. Flip coin with head prob = $1 - (1-1/t)^{se}$. If tails, continue to next edge
3. Choose random i in $[s_e]$ and set edge_res[i] = e_t.
4. Update G_res and tot_wedges.
5. Set q = new_wedges/tot_wedges
6. For each i in $[s_w]$ (for each entry in wedge_res)
    1. Flip coin with head prob. = q. If tails, continue.
    2. Pick uniform random w in NW
    3. Replace wedge_res[i] = w. Reset isClosed[i] = FALSE

Comments

- wedge_res doesn't need to store middle vertices of wedges. Store ends of wedges, indexed in hash table. So update to isClosed is fast.
- Updates to edge_res very rare!

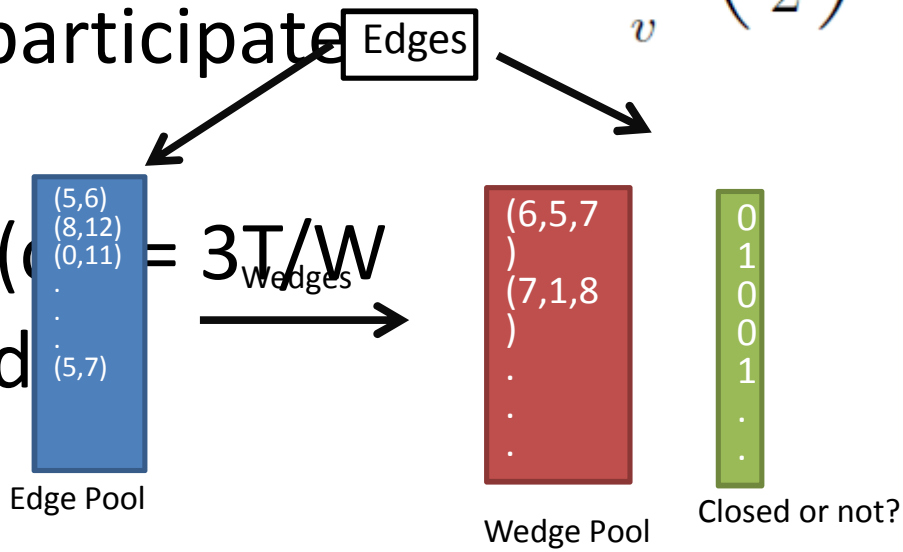
$$\sum_{t \leq m} 1 - (1 - 1/t)^{s_e} \approx \sum_{t \leq m} s_e/t \approx s_e \ln m$$

Sublinear techniques can be the key to face big data.

- If the data get bigger, the algorithms should get smarter.
  - Data will grow faster than computational resources.
  - By 2020, there will be 5.2TB/person.
- Sublinear algorithms maybe Sandia's signature in the area of Big Data.
  - They are effective.
  - We are better than everybody else.
- We have promising results on extending sublinear techniques to the streaming world.
  - We want to hear about your problems.

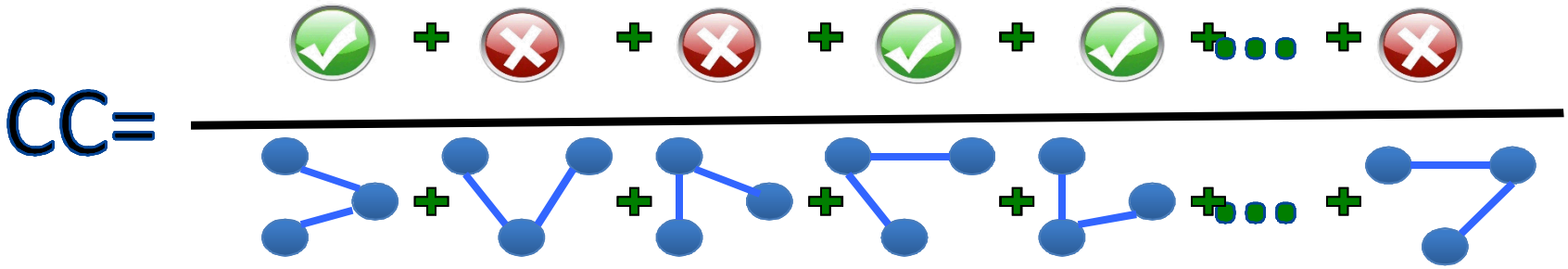
Triangles

- Graph G = (V,E). |V|=n, |E|=m.
- No. of triangles = T
- No. of wedges (paths of length 2) = W = $\sum_v \binom{d_v}{2}$
- Wedge is "closed" if it participate in triangle
- Global clustering coeff (= 3T/W = fraction of closed wed
- Amen.



Closed or not closed, that is the question.
W. Shakespeare

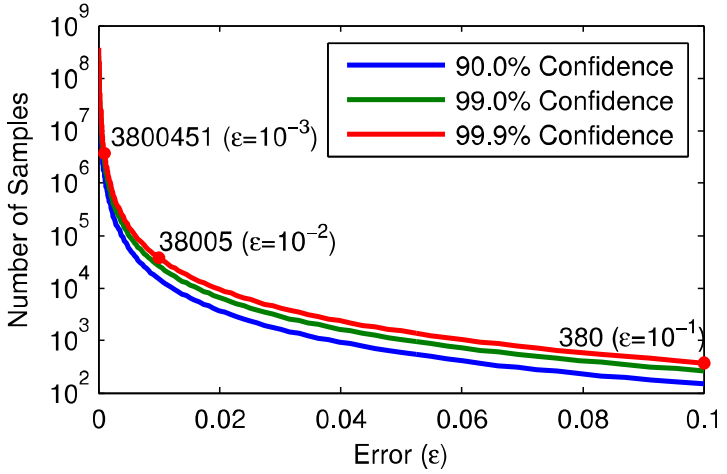
Sandia National Laboratories

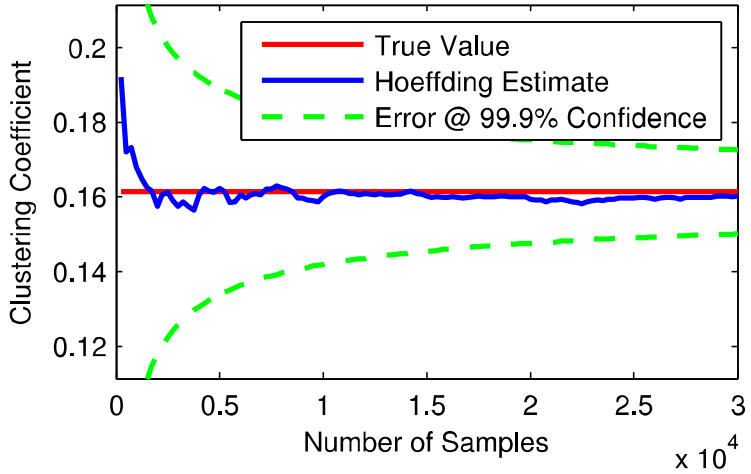


- Clustering coefficient (CC) is defined as the frequency of a wedge being closed.
- Can be considered as the success rate of an experiment with a binary outcome.
  - Each wedge is an experiment, which succeeds if it is closed, and fails otherwise.
- To estimate the frequency, we don't need to look at all the wedges, a small sample is enough.

Wedge-sampling provides provably accurate estimations

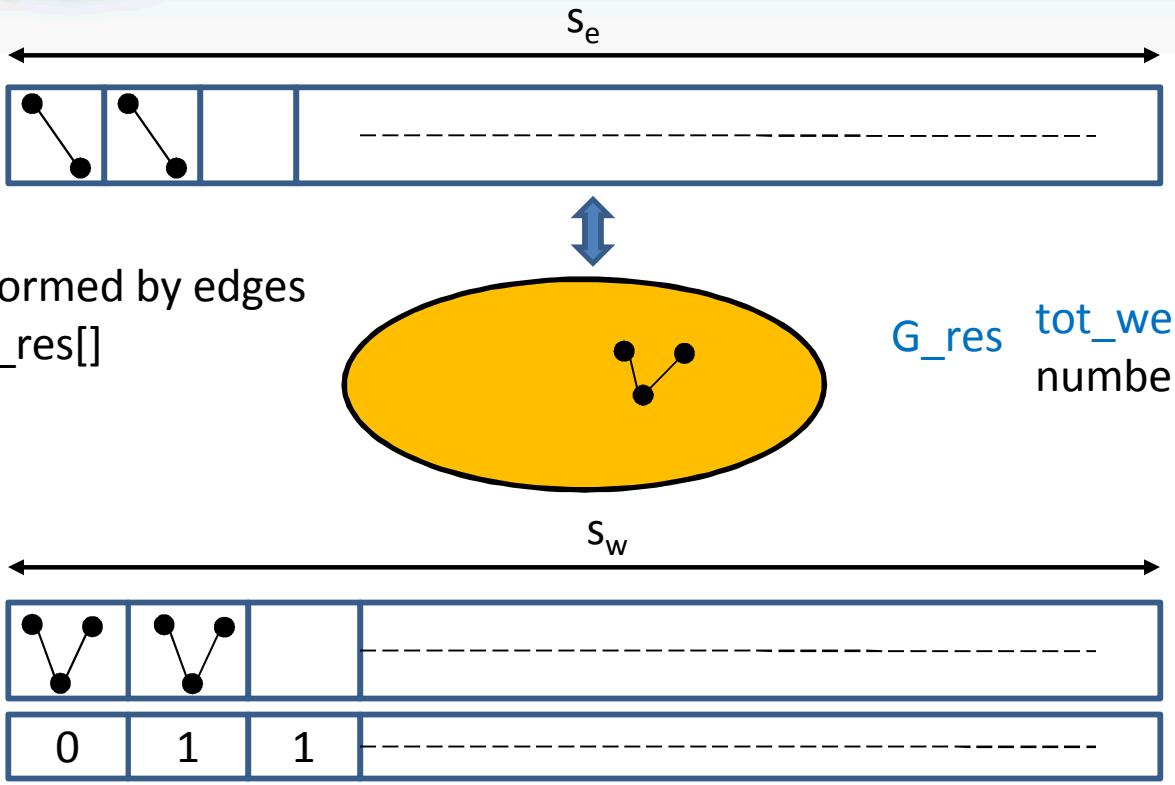
- Theorem: For error = ε and confidence = 1-δ, the number of samples required is $\left\lceil 0.5\varepsilon^{-2} \ln(\frac{2}{\delta}) \right\rceil$
- For 99.9% confidence and 1% error, we need only k = 38,005 samples





The number of samples in independent of the graph size.

Data stored by algorithm



- Parameters: the wedge and edge pool sizes
- Output:
  - Clustering coefficient: 3*fraction of closed entries
  - Triangle count: $pt^2/(s_e(s_e-1))$ X tot_wedges