Estimating Graph Properties through Sampling

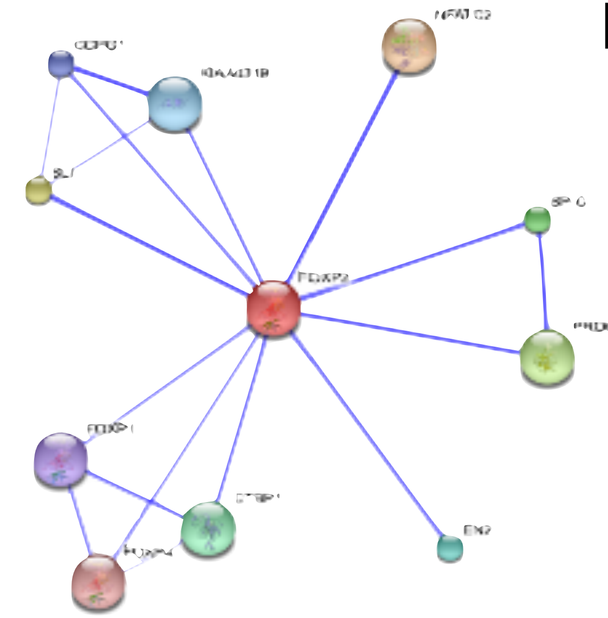
Shweta Jain
Advisor: Prof. C. Seshadhri

University of California,
Santa Cruz

Large Graphs

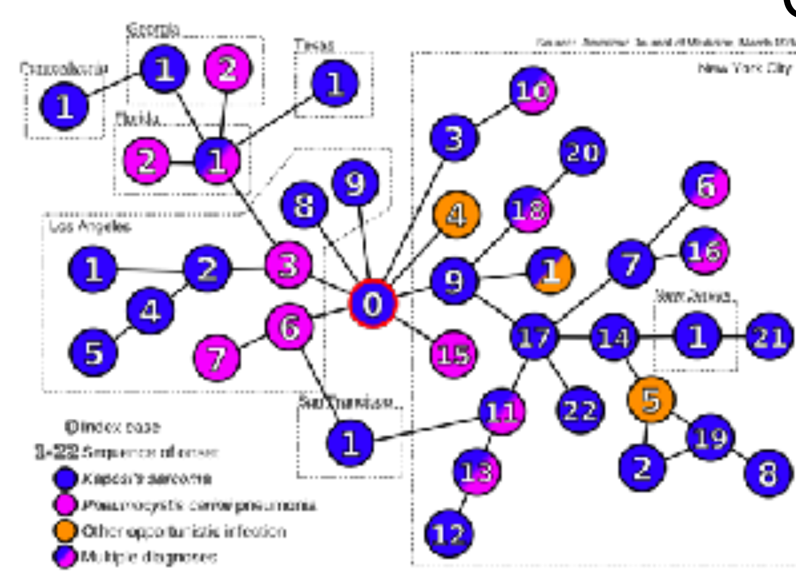
Social Network

Protein-Interaction Networks



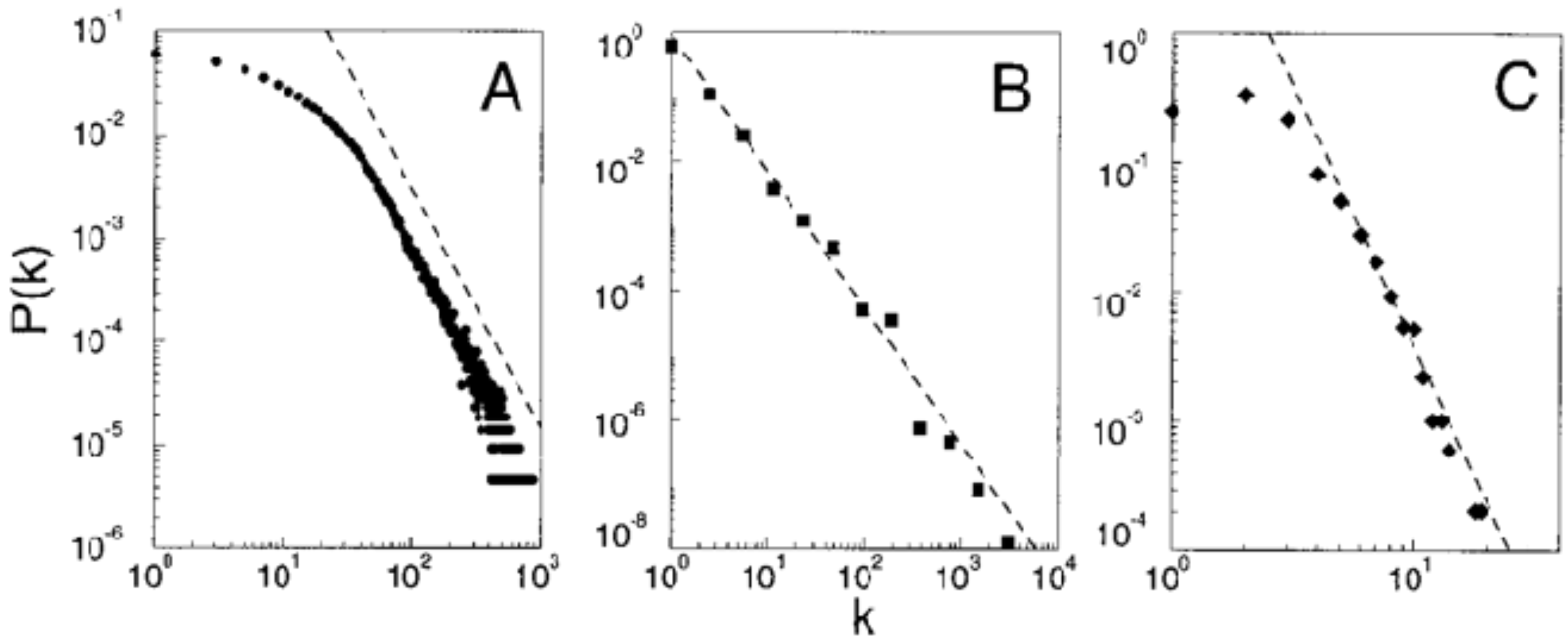
Routing Networks

Citation Networks



Peculiarities of real-world graphs

- Degree distribution
  - Heavy tailed

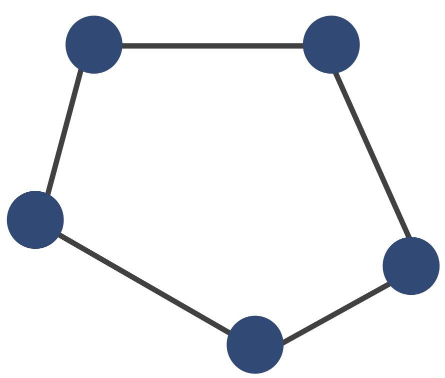


A: Actor collaboration network, B: WWW, C: Power Grid data [Barabási et. al., 1999]

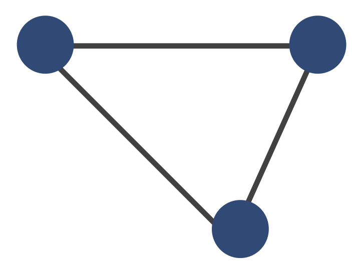
Source: www.sciencemag.com

Peculiarities of real-world graphs

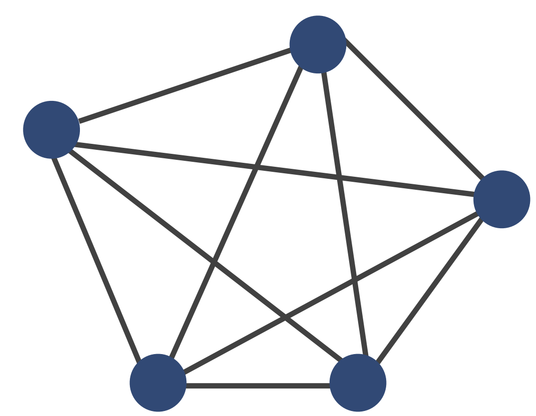
- Counts of patterns: cycles, triangles, cliques



5-cycle



3-clique
(triangle)



5-clique

- Avg. distance between nodes - small world property
- High clustering coefficients

Need for graph sampling

- Scale - traditional graph-theoretic algorithms impractical
- Limitations of access model e.g. streaming
- Can utilize unique characteristics of real-world graphs

Goals

- Estimate global characteristics from small sample.
- Fast, work well on real-world instances.
- Accurate, with provable error bounds.

Applications

- Computationally hard problems - clique counting
- Restricted access model - estimating the degree distribution

A Fast and Provable Method for Estimating Clique Counts using Turán’s Theorem.

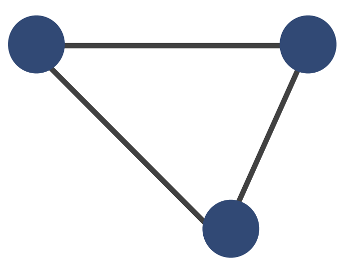
Shweta Jain
C. Seshadhri

University of California, Santa Cruz

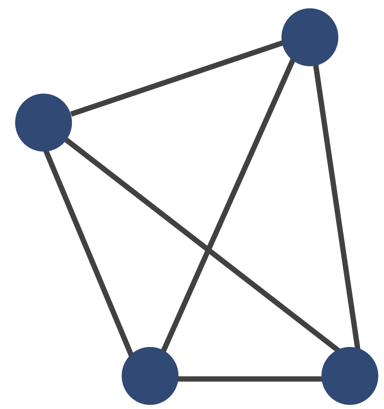
WWW 2017 Best Paper

Cliques

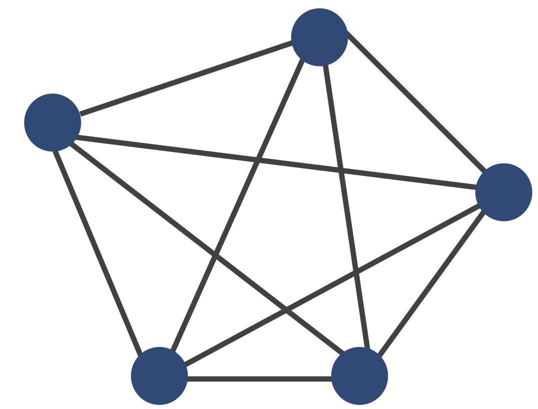
- k-clique: set of k vertices all connected to each other.



3-clique
(triangle)



4-clique



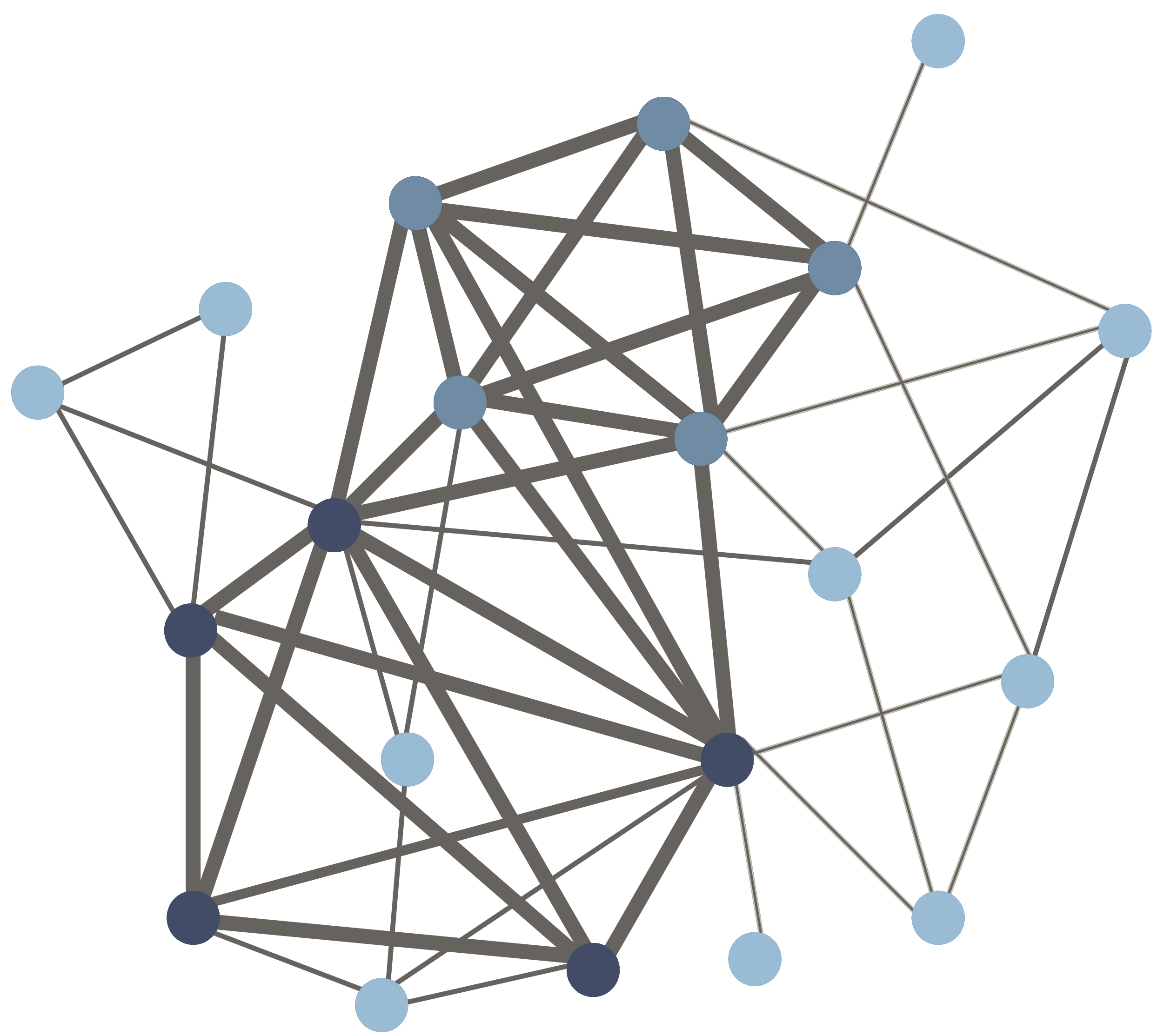
5-clique

- [Holland et. al., 1970], [Milo et. al., 2002], [Burt, 2004], [Przulj et. al., 2004], [Hanneman et. al., 2005], [Hormozdiari et. al., 2007], [Faust, 2010], [Jackson, 2010], [Tsourakakis et. al., 2015], [Sizemore et. al., 2016] - clique counts appear in all these papers.
- Used in modeling, community detection, spam detection etc.

Problem Statement

- Given a simple, undirected graph G, and a positive integer k, estimate the number of k-cliques in G.

#5-Cliques = [illegible]



Prior theoretic

theo work

- Clique counting:
  - Arboricity and subgraph listing algorithms. [Chiba et. al., 1985]
  - Finding dense subgraphs with size bounds. [Alon et. al., 1994]
  - Efficient algorithms for clique problems. [Vassilevska, 2009]
- Maximal clique counting:
  - Finding all cliques of an undirected graph. [Bron et. al., 1963]
  - Worst case time complexity of generating all maximal cliques. [Tomita et.al., 2004]
  - Listing all maximal cliques in large sparse real-world graphs. [Eppstein et. al, 2013]

Challenge

- Combinatorial explosion!

GRAPH	VERTICES	EDGES	7-CLIQUES	10-CLIQUES
web-BerkStan	0.6M	6M	**9T**	**50000T**
as-skitter	2M	11M	**73B**	**22T**
com-lj	4M	34M	**510T**	**14000000T**
com-orkut	3M	110M	**360B**	**31T**

$\underbrace{\hspace{6em}}$

Enumeration is costly.
Hence, approximate.

Practical approaches

- Practical approaches:
  - Color Coding [Alon et. al, 1994], [Hormozdiari et. al., 2007], [Betzler et. al., 2011], [Zhao et. al., 2012]
  - Edge Sampling, GRAFT [Tsourakakis et. al., 2009], [Tsourakakis et. al., 2011], [Rahman et. al., 2014]
  - MCMC based [Bhuiyan et. al., 2012]
  - Parallel algorithm using MapReduce [Finocchi et. al., 2015]
  - kClist [Danisch et. al., 2018]

Our contribution

- We present a randomized algorithm, **TuránShadow** that approximates the number of k-cliques in G and has the following properties:
  - Runs on a single machine
  - Provable error bounds

Our contribution

- Extremely fast and accurate

GRAPH	7-CLIQUES	TIME	ERROR %
web-BerkStan	9.3T	< 4 minutes	1.05
as-skitter	73B	< 3 minutes	0.23
com-orkut	361B	< 2 hours	1.97

- For 10 cliques, **no other method terminated** for all graphs in **min{100xTuranShadow, 7 hours}**!

Main theorem

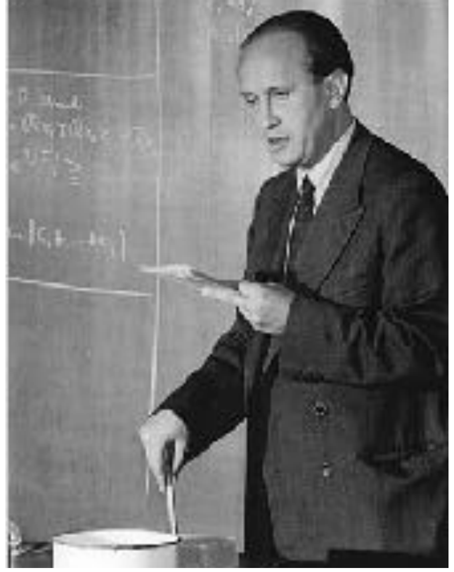
Let **S** be the **Turán k-clique shadow** of G. Then w.h.p. TuránShadow outputs a $(1 \pm \epsilon)$-approximation to the number of k-cliques in G.

The running time of TuránShadow is $O^*(\alpha|\mathbf{S}|+m+n)$.

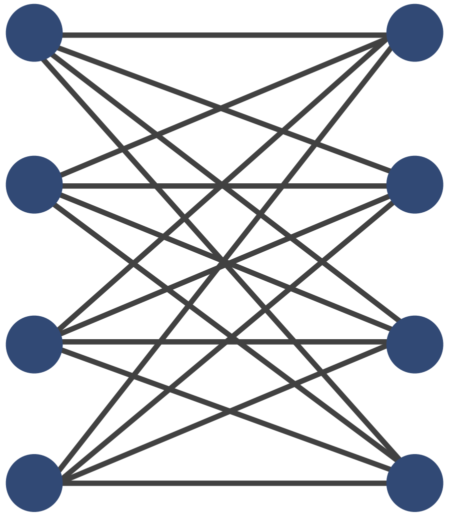
α: degeneracy
m: #edges
n: #vertices

Degeneracy

- α: degeneracy of graph
- Measure of density, low for real-world graphs
- Let T: set of all subgraphs of G
- Degeneracy = $\max_{t \in T} \min_{v \in t} \{\text{degree of } v \text{ in } G_{|t}\}$

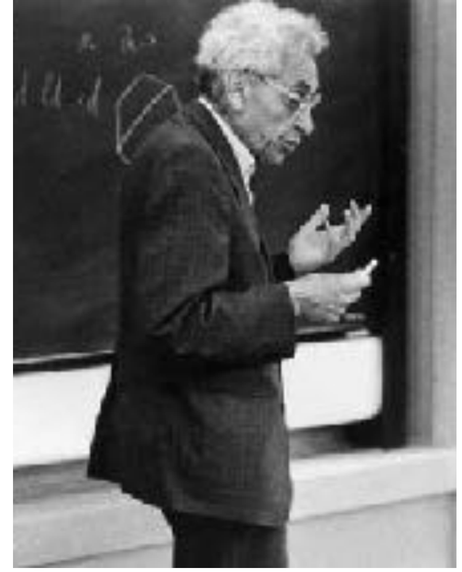


How many edges can a n-vertex graph have without having a triangle?

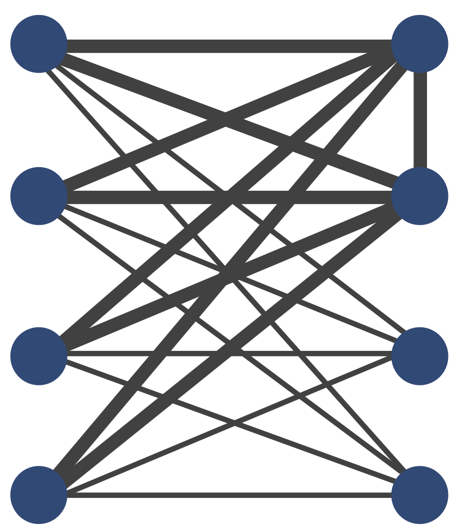


Ans: $\dfrac{n^2}{4}$

[Turán, 1941] If the graph has more than $\dfrac{n^2}{4}$ edges, then it **must** have a triangle.



[Erdös, 1941] If the graph has even **one** more edge than $\frac{n^2}{4}$, then it must have $\Omega(n)$ triangles.



density = $\frac{\#edges}{\binom{n}{2}}$

$= \frac{1}{2}$

Thus, if density > $\frac{1}{2}$, then graph necessarily has $\Omega(n)$ triangles.

Turán's theorem

Generalizes for larger k.

If a graph on n vertices has density greater than

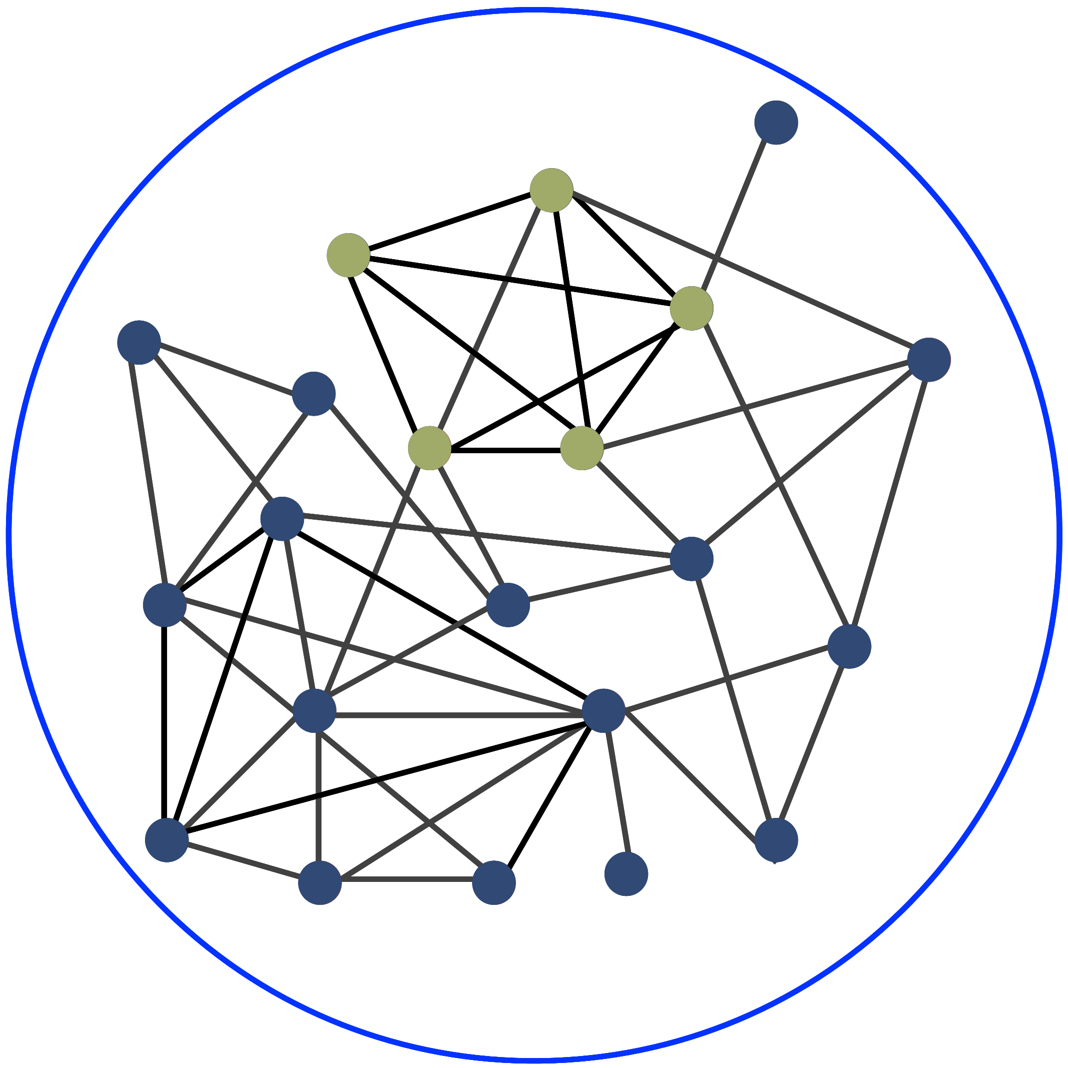
$$1 - \frac{1}{k-1}$$

then it must have

$$\Omega(n^{k-2})$$

k-cliques.

Naïve algorithm



G

n = 1M

k = 5

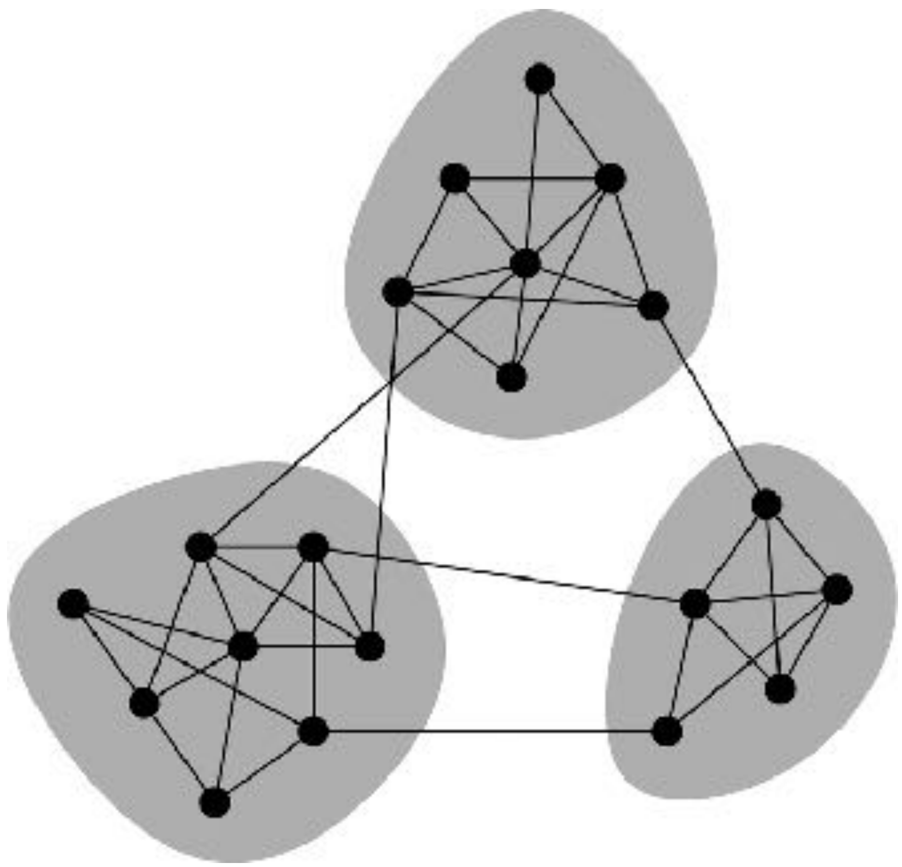
#5-cliques = 100T

$$E[\#samples] = \frac{\binom{1M}{5}}{\#5\text{-}cliques}$$

$$\cong 10^{16}$$

Key Idea

Real world graphs have dense pockets.



Drill down on dense pockets and count cliques within them!

Turan Shadow

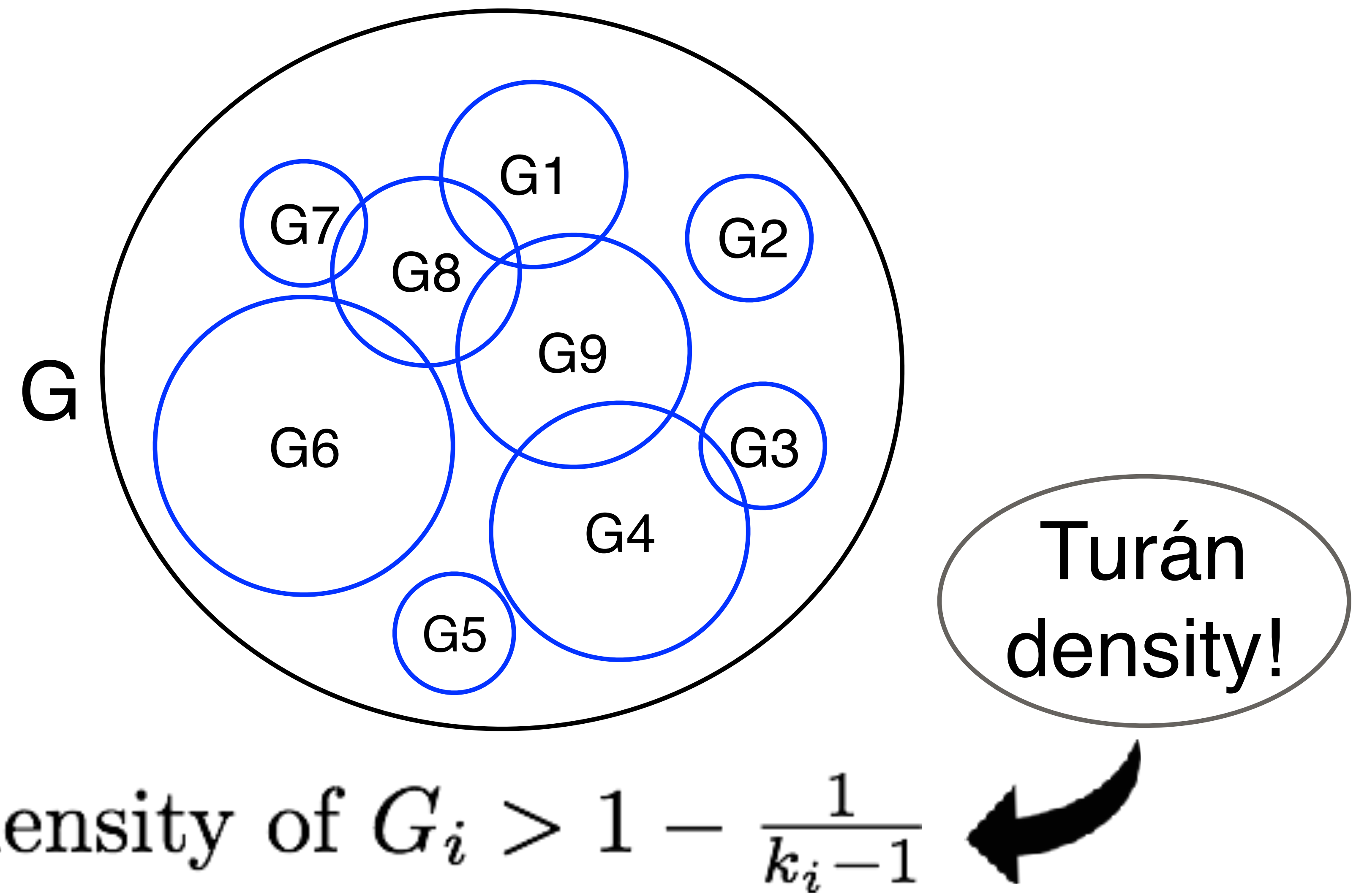
decompose

G-> G1, k1

G2, k2

G3, k3

…

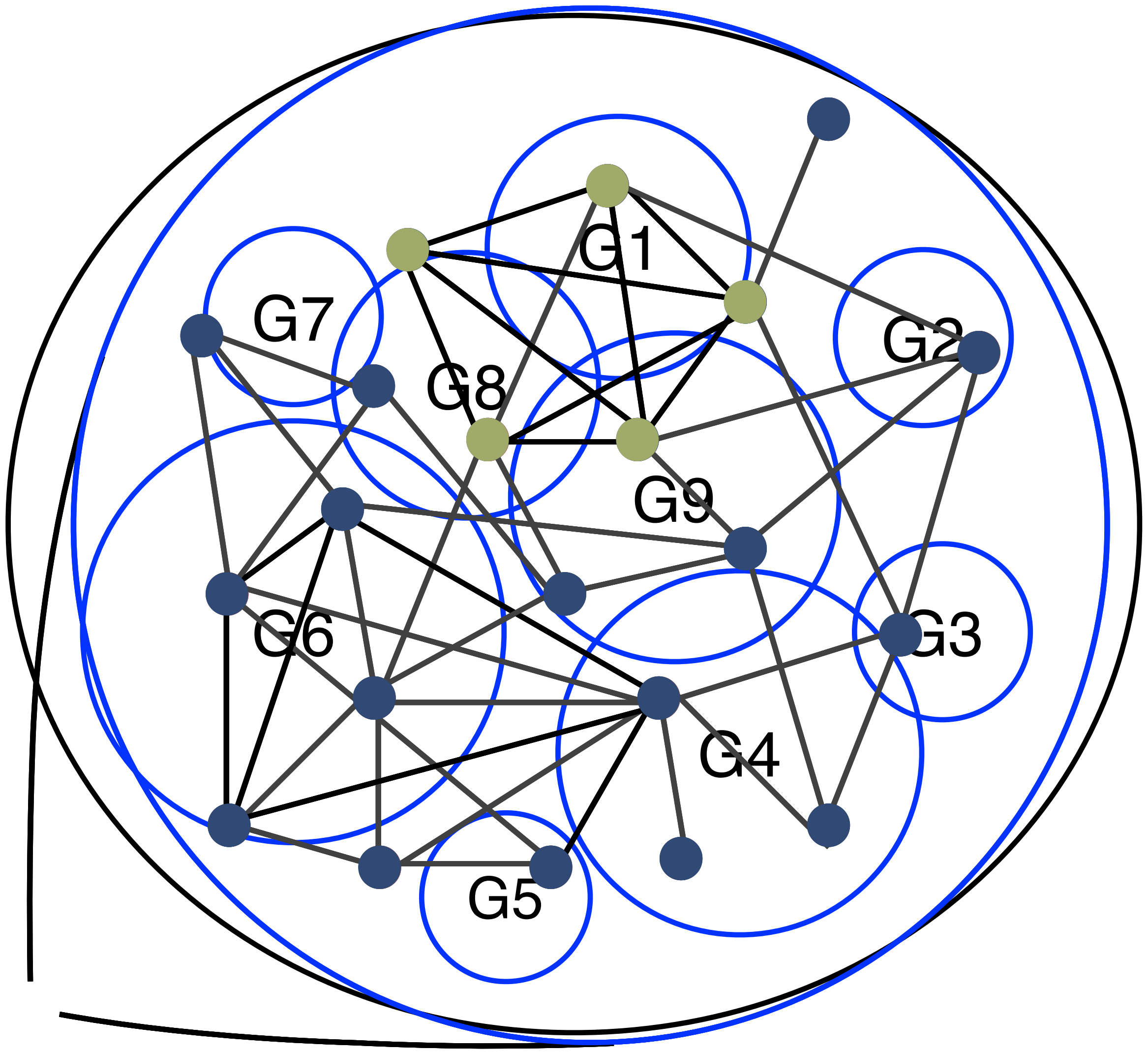


$$\text{density of } G_i > 1 - \frac{1}{k_i - 1}$$

$$\#k\text{-cliques in } G = \sum_i \#k_i\text{-cliques in } G_i$$

$$\textbf{Revised goal: Estimate } \#k_i\textbf{-cliques in } G_i$$

Turan Shadow



G1

n1= 21

k1= 5

C = 0

#samples = 3

$$\text{E[\#samples]} = \frac{\binom{n1}{5}}{\#5\text{-}cliques}$$

Erdös!

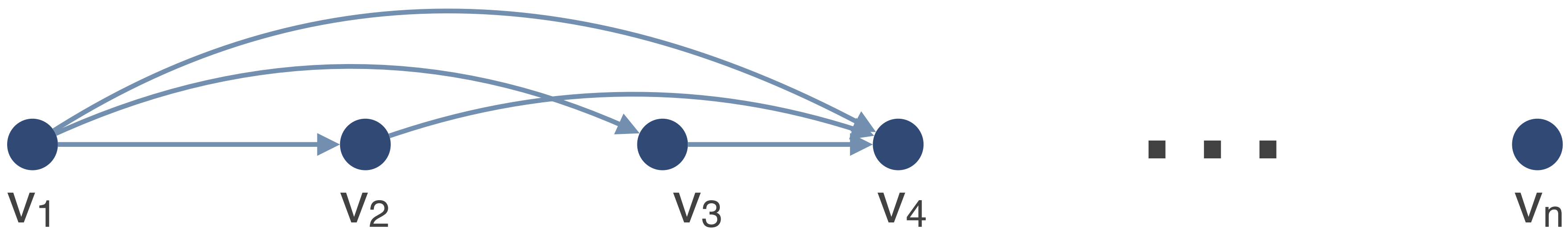
$$< \frac{\binom{n1}{5}}{n1^3}$$

$$< n1^2$$

Constructing the shadow

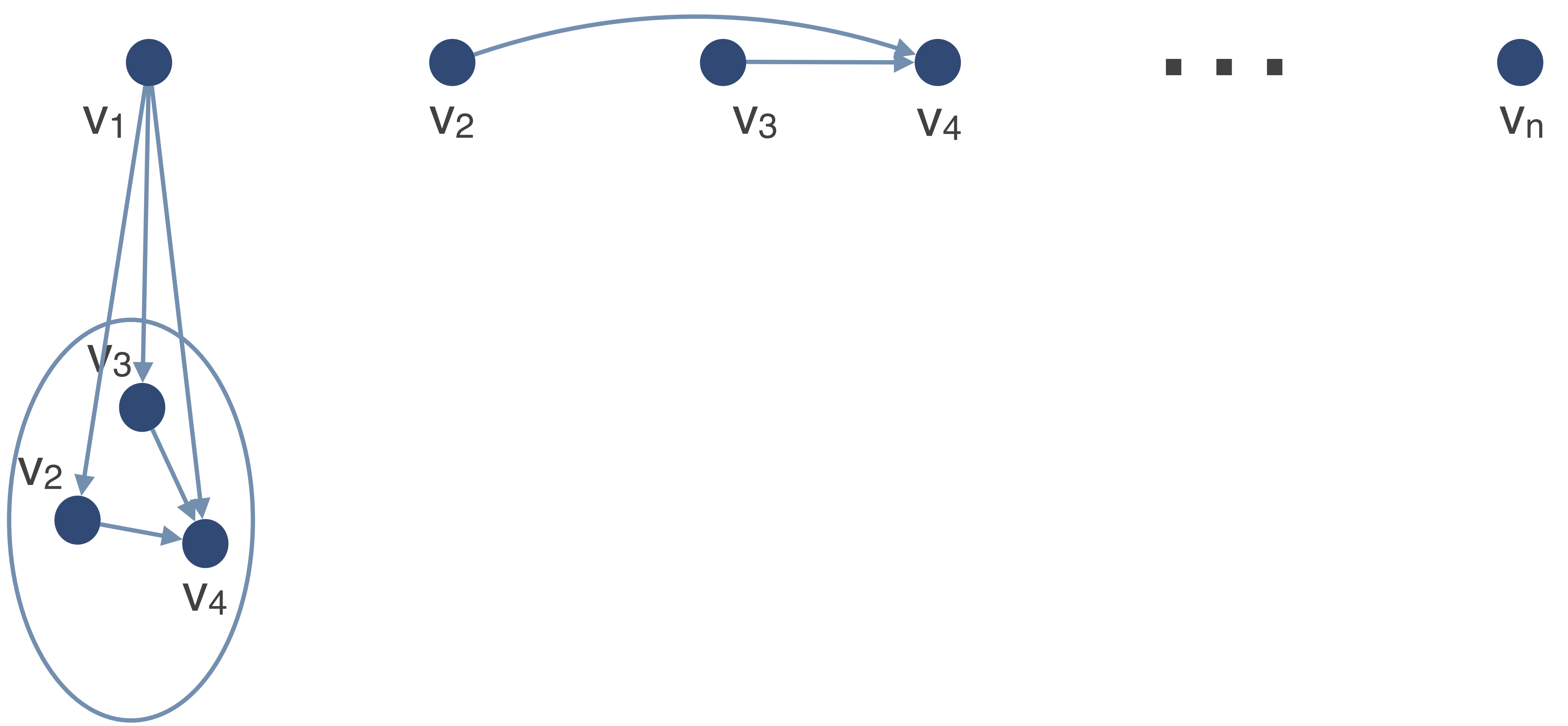
- Convert G to a DAG - order by degeneracy
- Build clique enumeration tree, stopping whenever Turán density is reached.

Constructing the shadow



Convert G to DAG

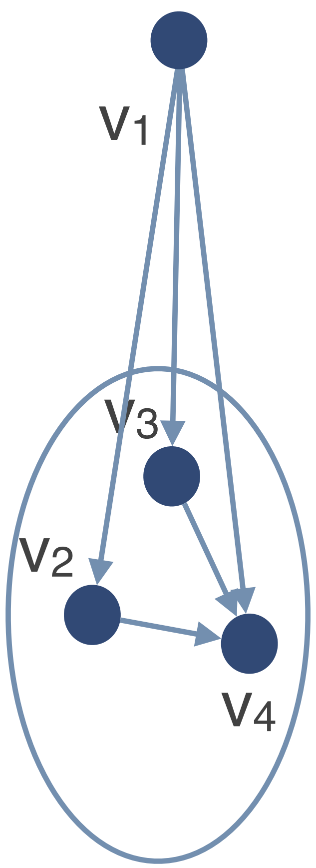
Constructing the shadow



Convert G to DAG

Check outnbrhd of v_1

Constructing the shadow



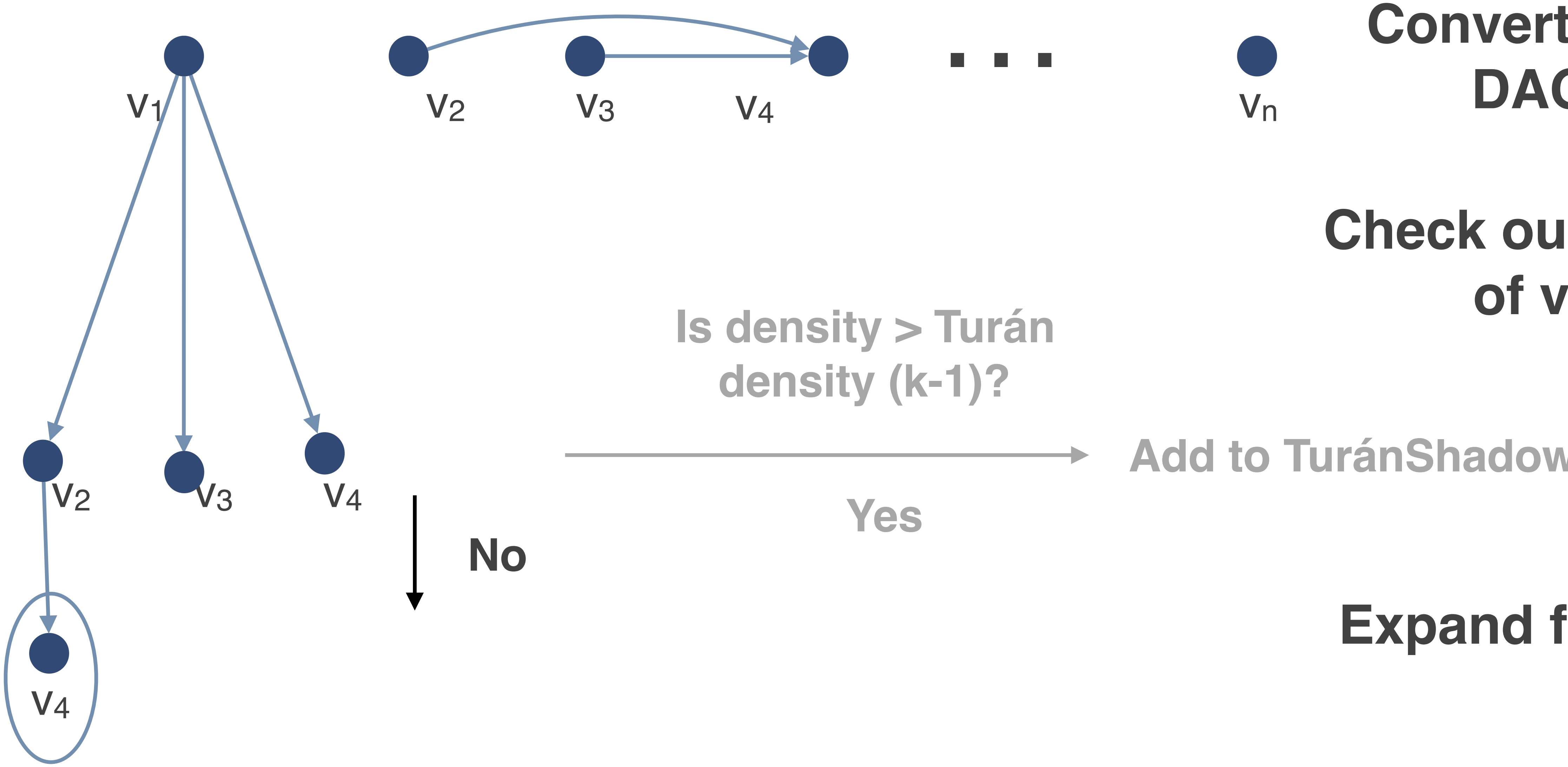
Convert G to DAG

Check outnbrhd of v_1: $\Gamma^+(v_1)$

Is density > Turán density (k-1)?

Yes → **Add to TuránShadow**

Constructing the shadow



$\Gamma^+(v_1) \cap \Gamma^+(v_2)$

Convert G to DAG

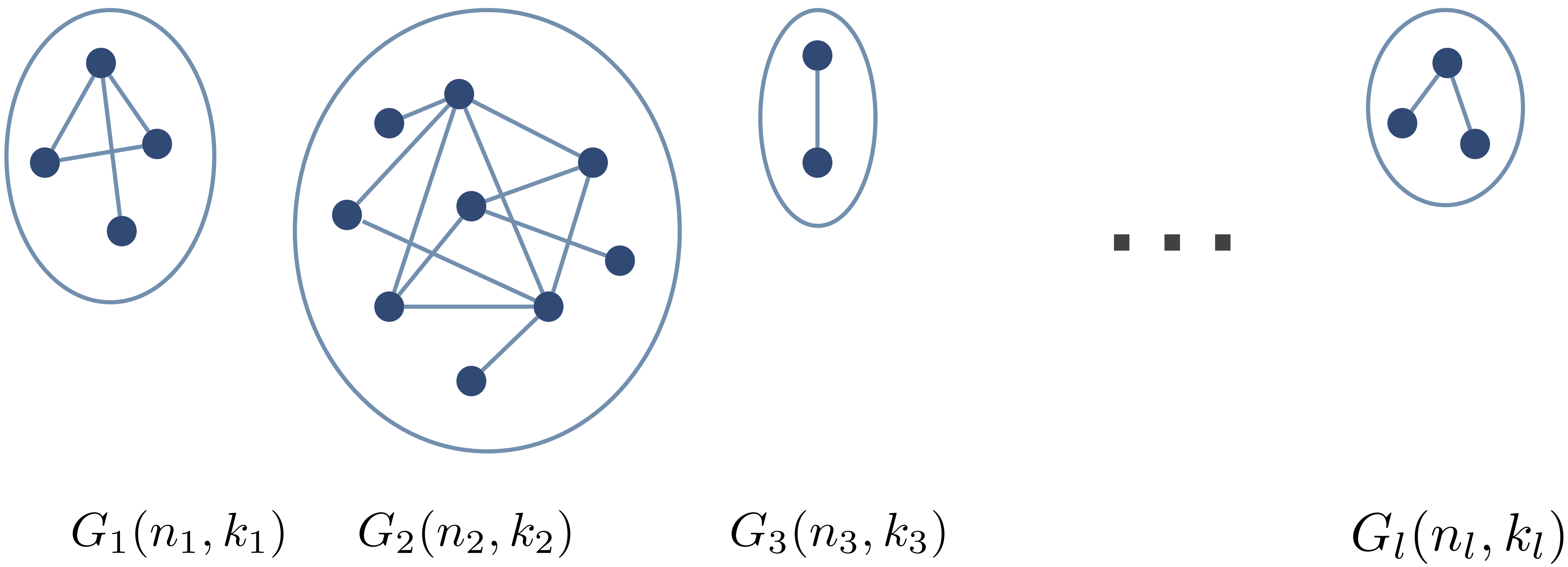
Check outnbrhd of v_1

Expand further

Sampling

Sample leaf i with probability $\dfrac{\binom{n_i}{k_i}}{\sum_{j \leq l} \binom{n_j}{k_j}}$

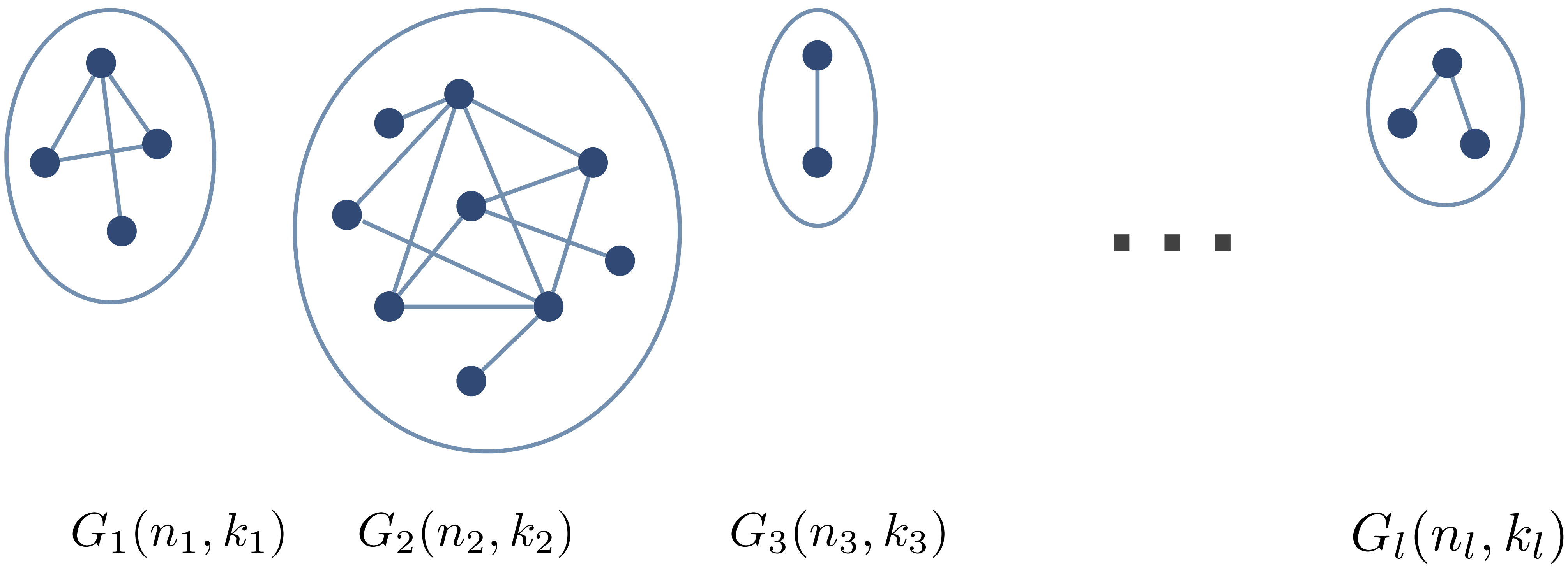
Randomly sample k_i vertices from leaf i



Sampling

Bernoulli r.v. $X = 1$ if k_i-clique, else 0

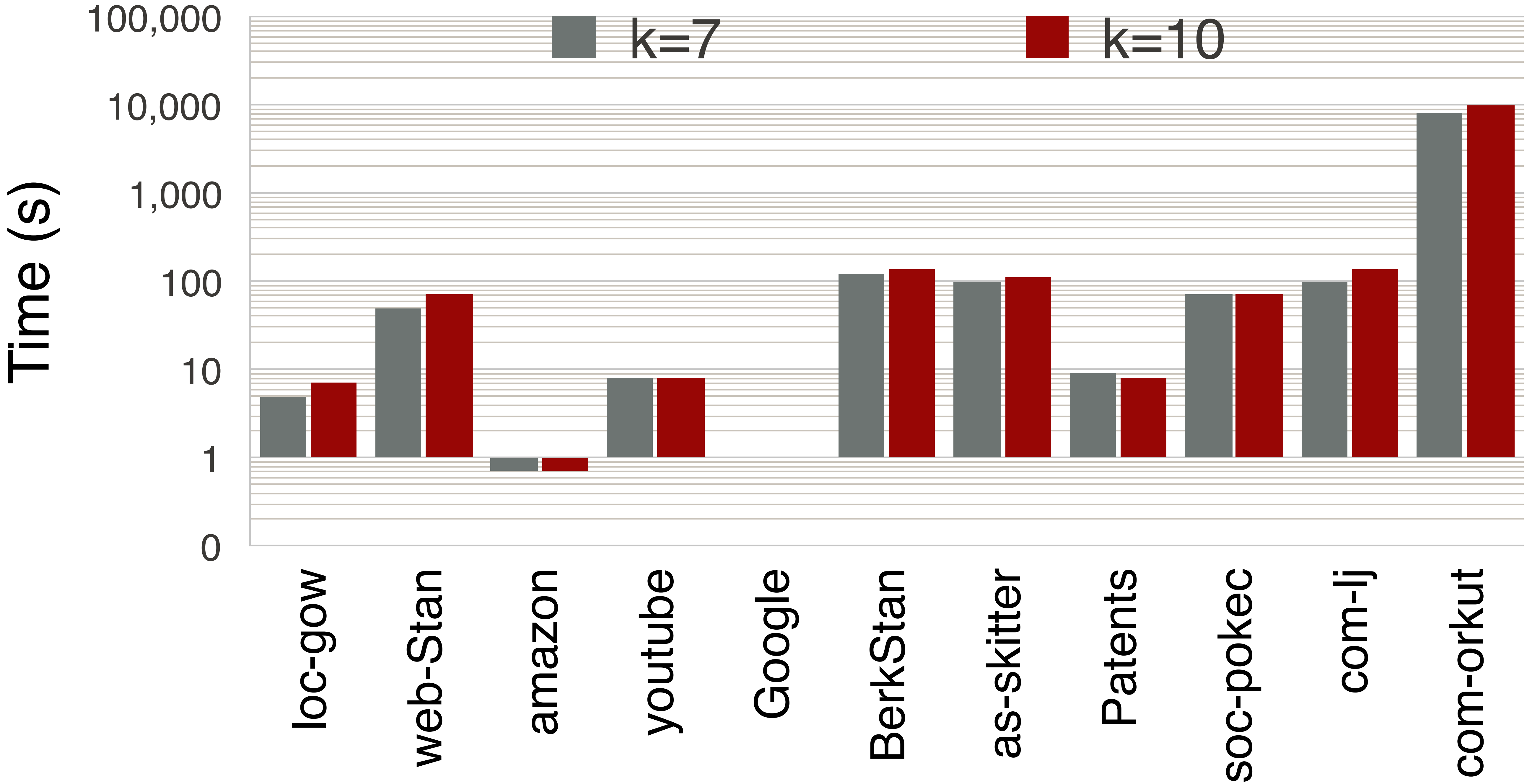
$$Exp[X] = \frac{\#k\text{-cliques in G}}{\sum_{j \leq l} \binom{n_j}{k_j}}$$



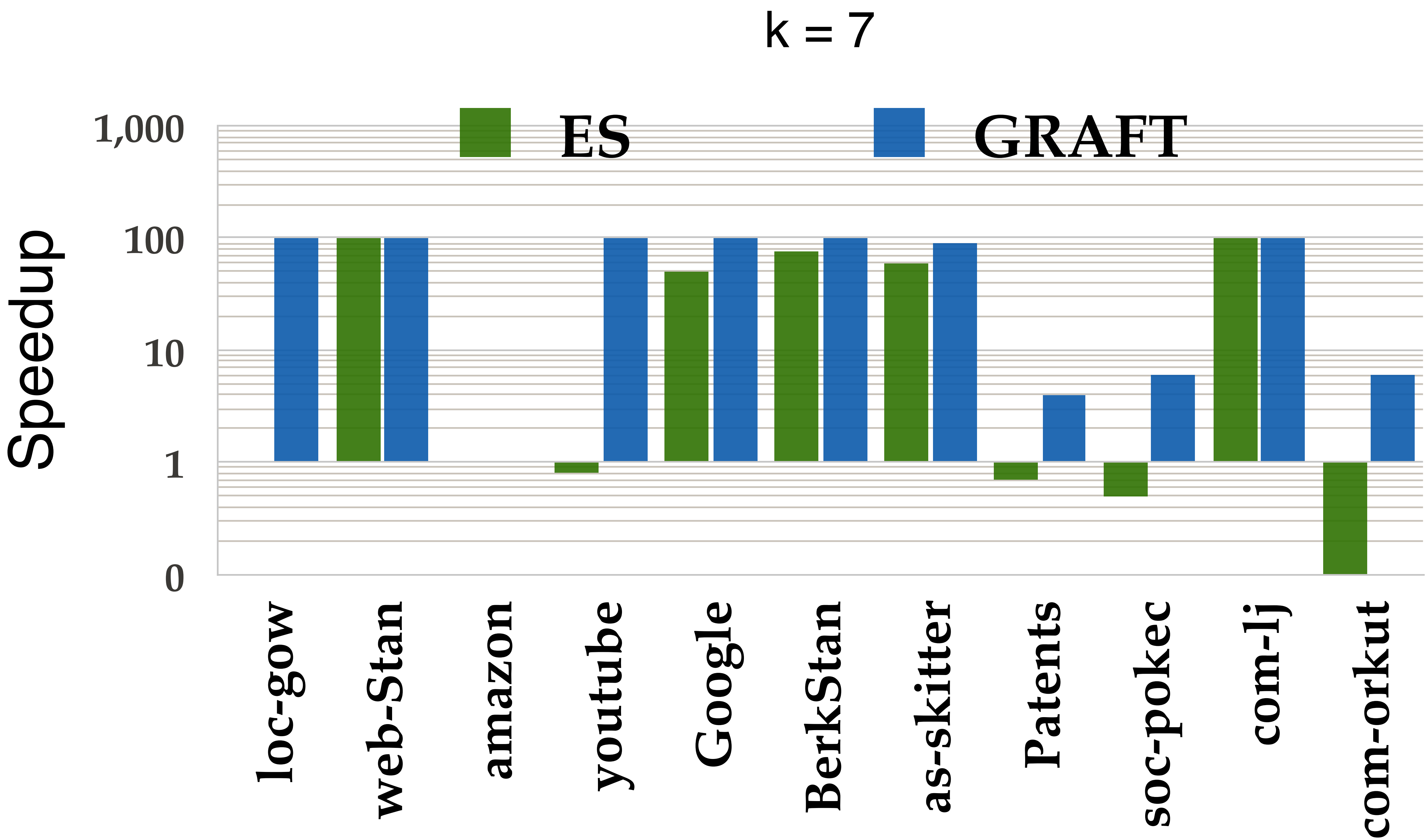
Putting it all together

- Construct Turán Shadow
- Setup distribution over leaves
- Sample from distribution and scale success ratio

7 and 10 Clique Count Estimation Performance



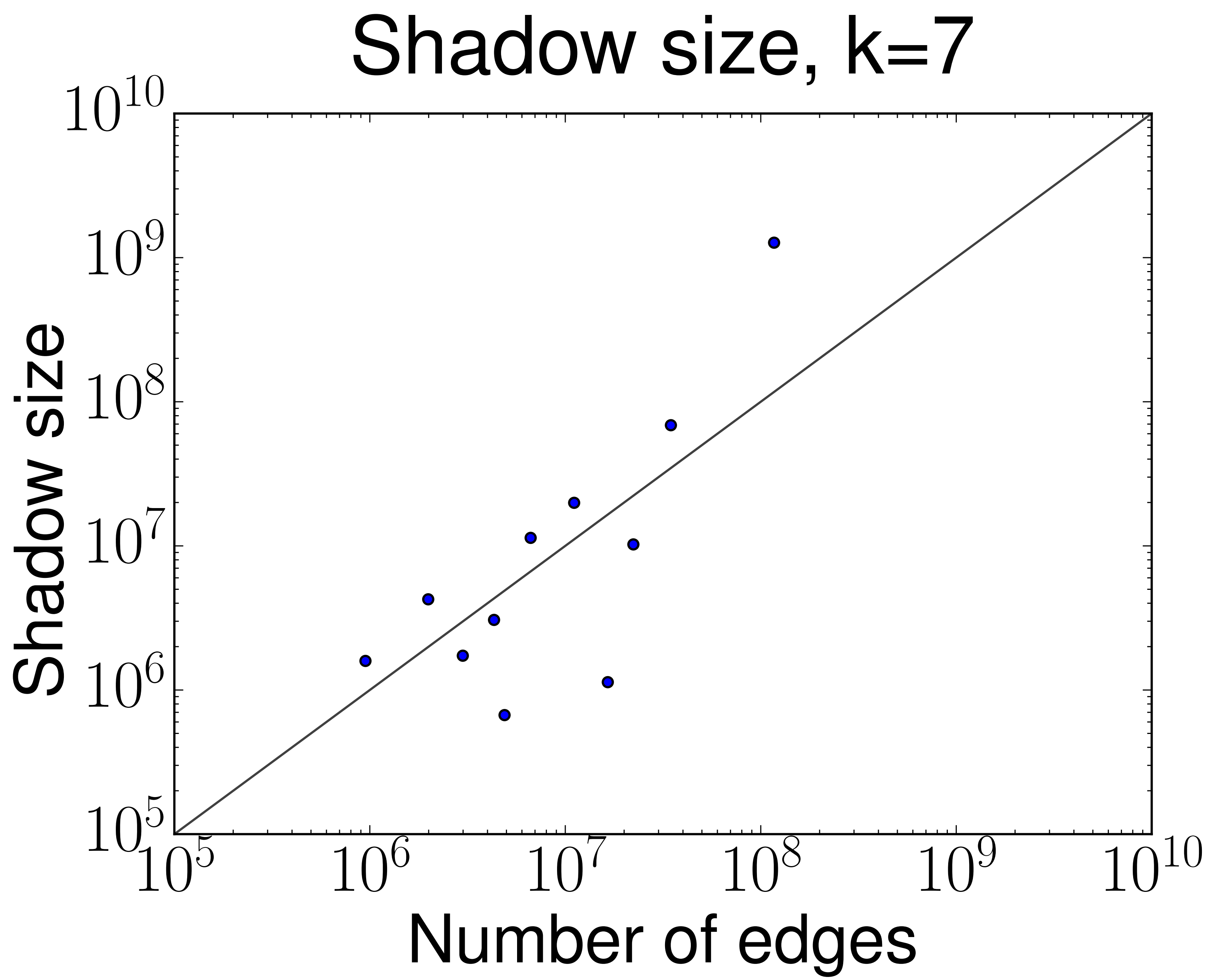
TuranShadow terminated in **minutes** for all graphs except com-orkut (3M/100M) for which it took 3 hours.



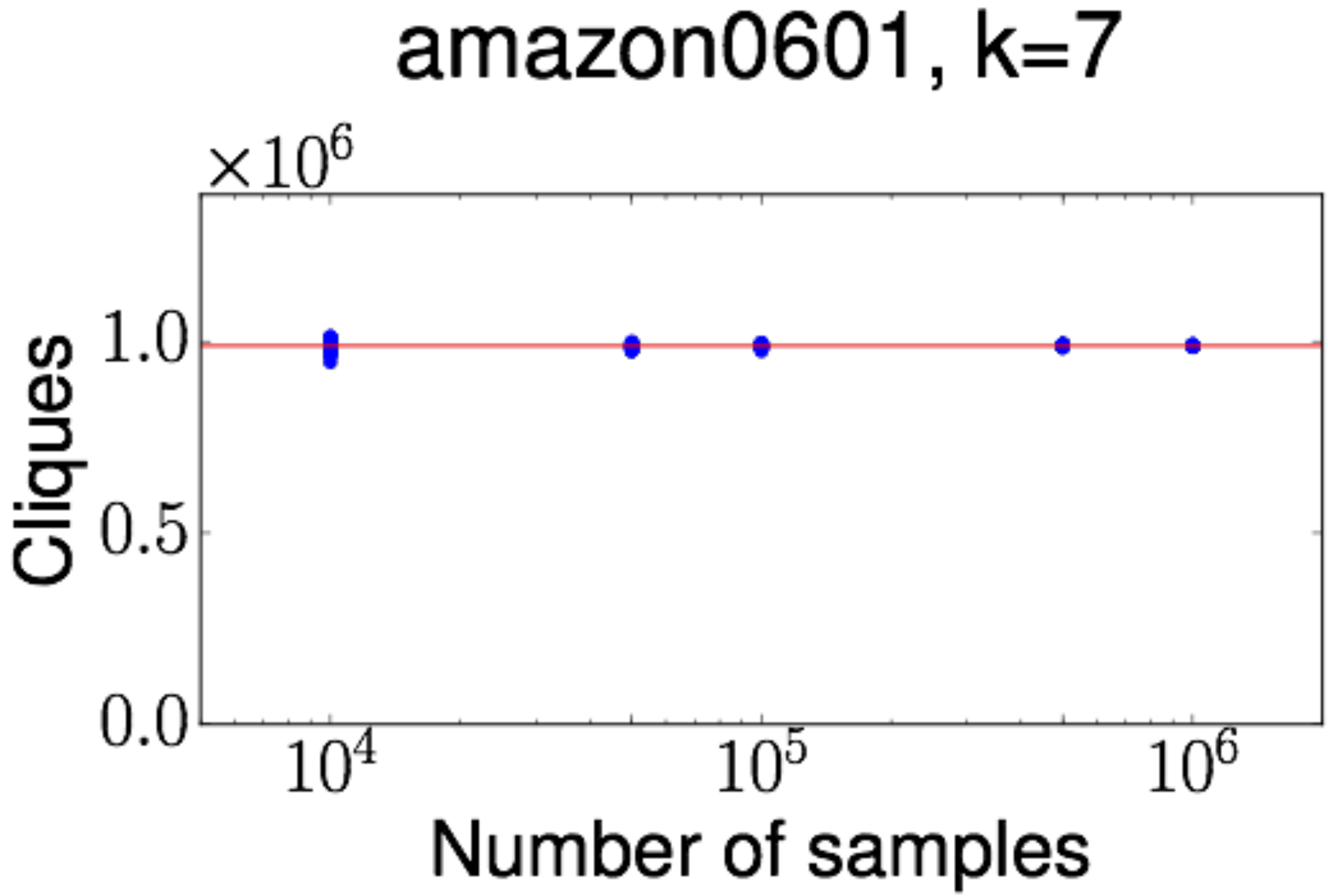
3-100x speedup for k=7.

For k=10, **no other algorithm terminated** for all graphs in min{100x, 7 hours}

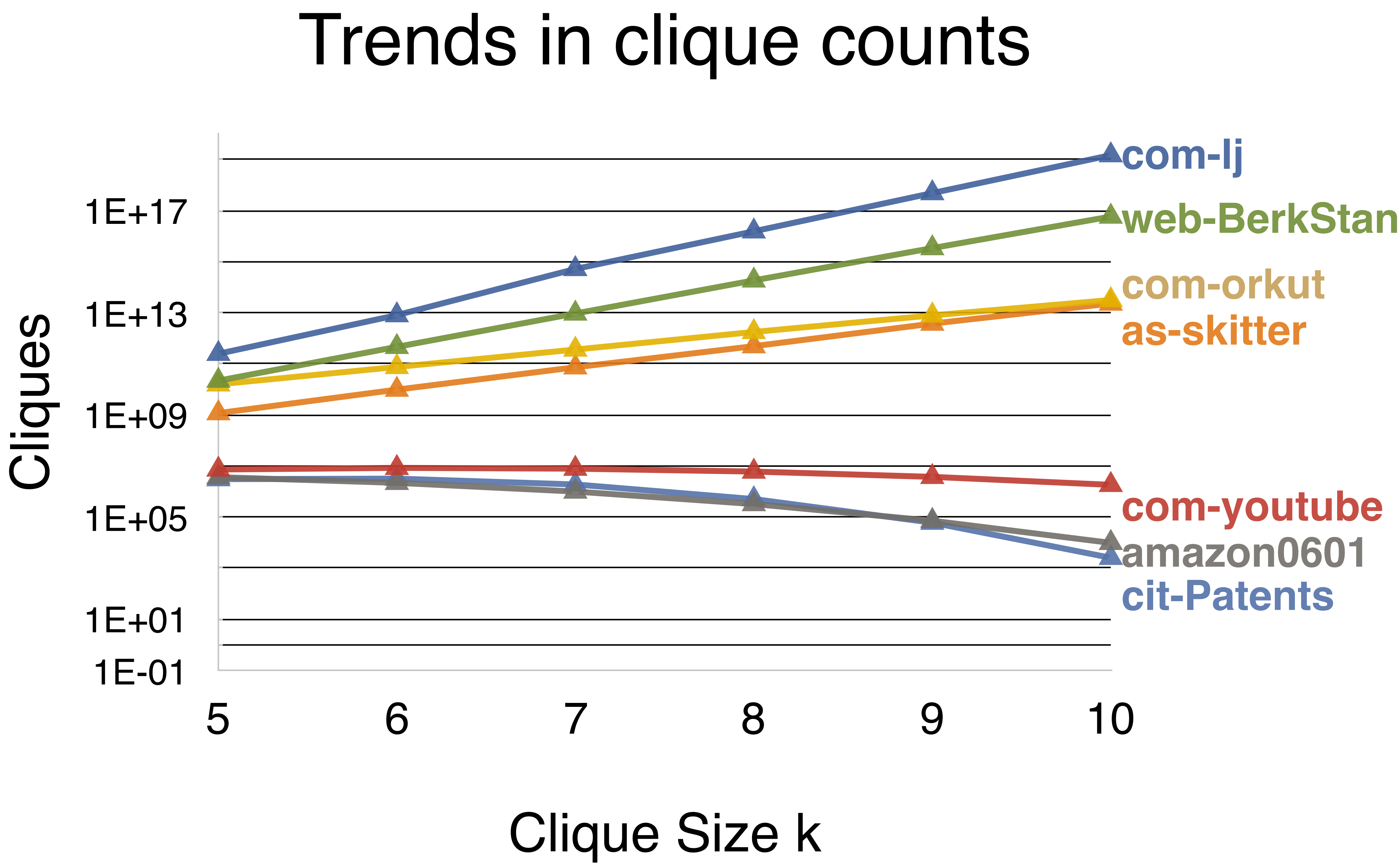
Size of shadow



Shadow size roughly linear in m.



Less than 2% error with just 50,000 samples.


Trends in clique counts
Cliques
1E+17
1E+13
1E+09
1E+05
1E+01
1E-01
5
6
7
8
9
10
Clique Size k
com-lj
web-BerkStan
com-orkut
as-skitter
com-youtube
amazon0601
cit-Patents

What we achieved

- We make clique-counting feasible for larger cliques.
- Single commodity machine. No need to use MapReduce.
- Extremely fast and accurate
- Provable error bounds

Open Questions

- Feasible for cliques of size $k > 10$?
- Can we count near-cliques?
- Can this approach be used for dense subgraph discovery?

Thank you

Questions?

Provable and Practical Approximations For the Degree Distribution using Sublinear Graph Samples*

WWW 2018

Talya Eden
Tel Aviv University

Shweta Jain
University of California, Santa Cruz

Ali Pinar
Sandia National Labs

Dana Ron
Tel Aviv University

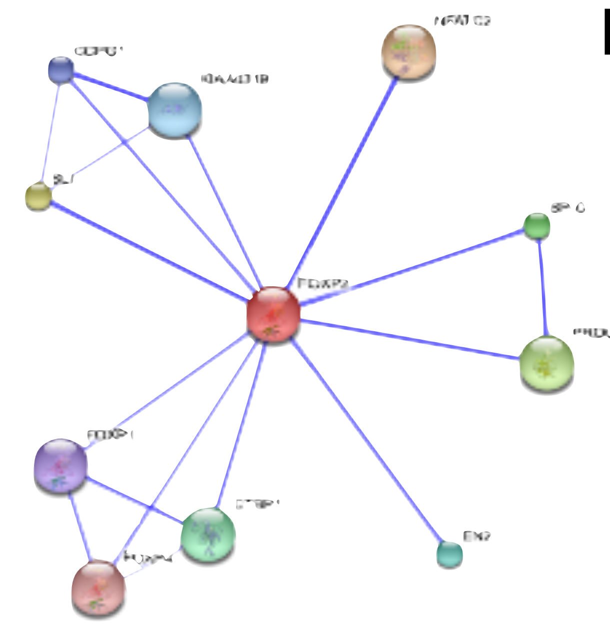
C. Seshadhri
University of California, Santa Cruz

* Talya and Shweta are equal contributors.

Large Graphs

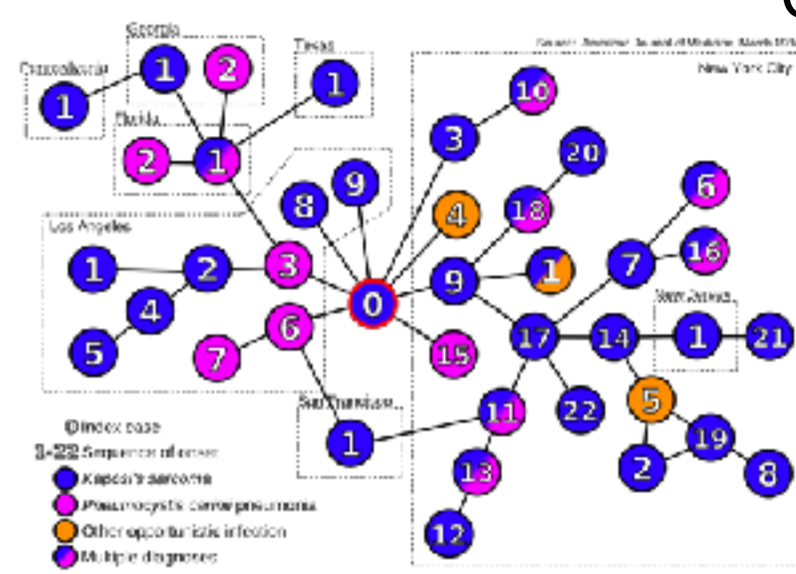
Social Network

Protein-Interaction Networks



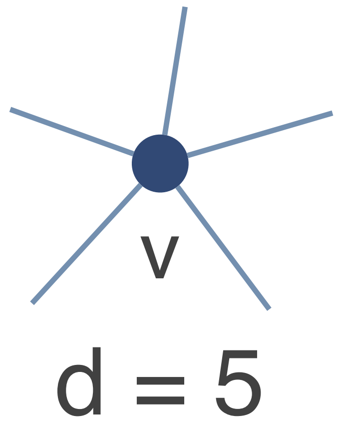
Routing Networks

Citation Networks



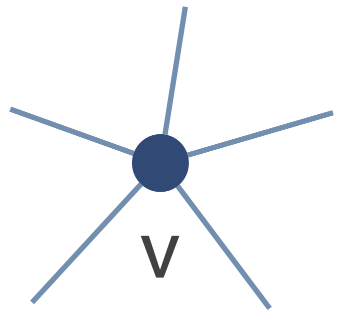
Degree Distribution

- Degree(v) = #vertices v is connected to



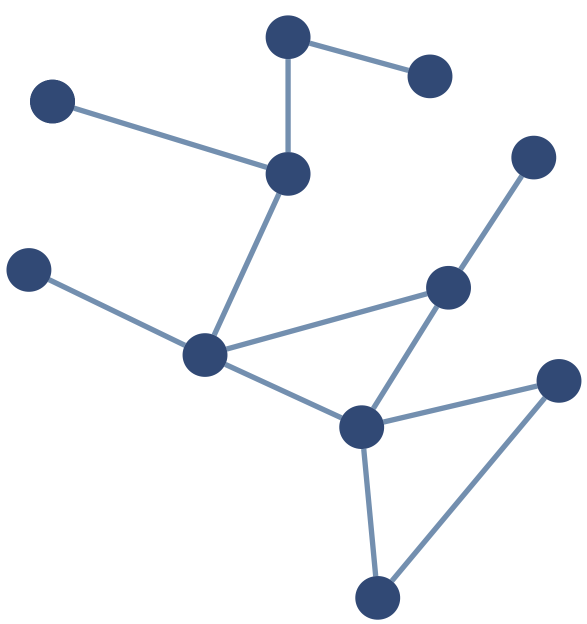
Degree Distribution

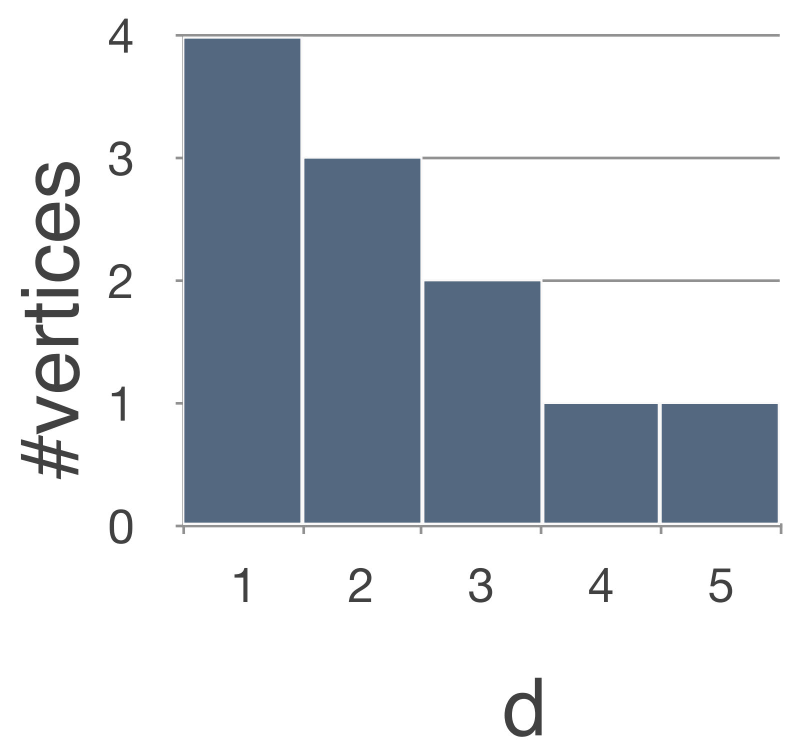
- Degree(v) = #vertices v is connected to



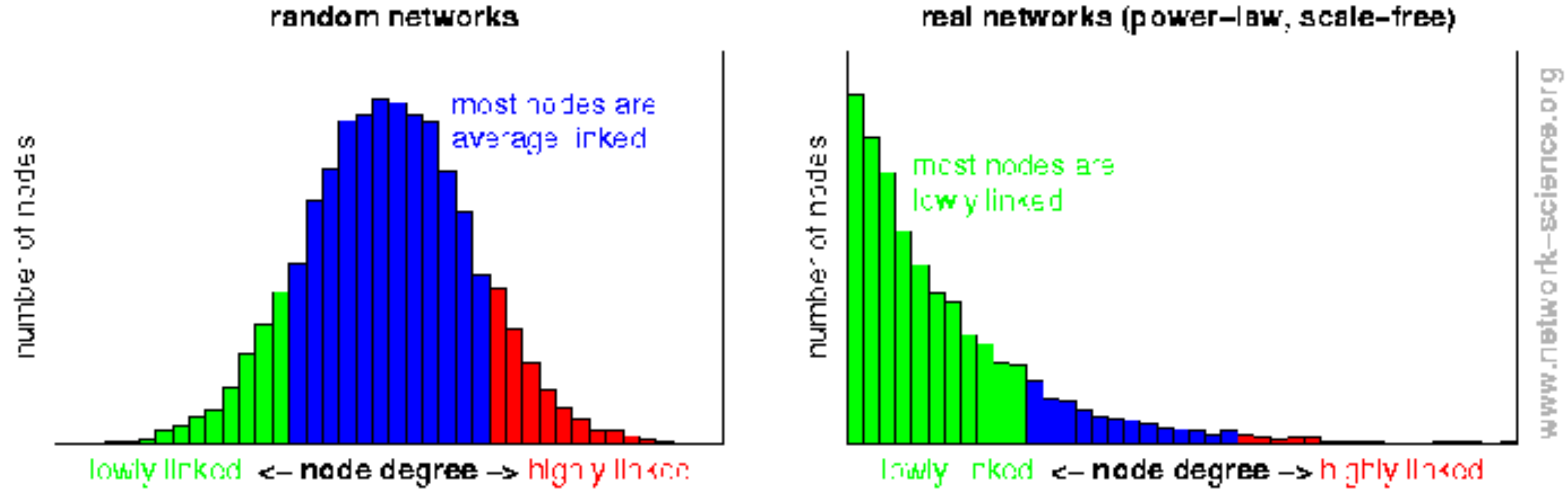
d = 5

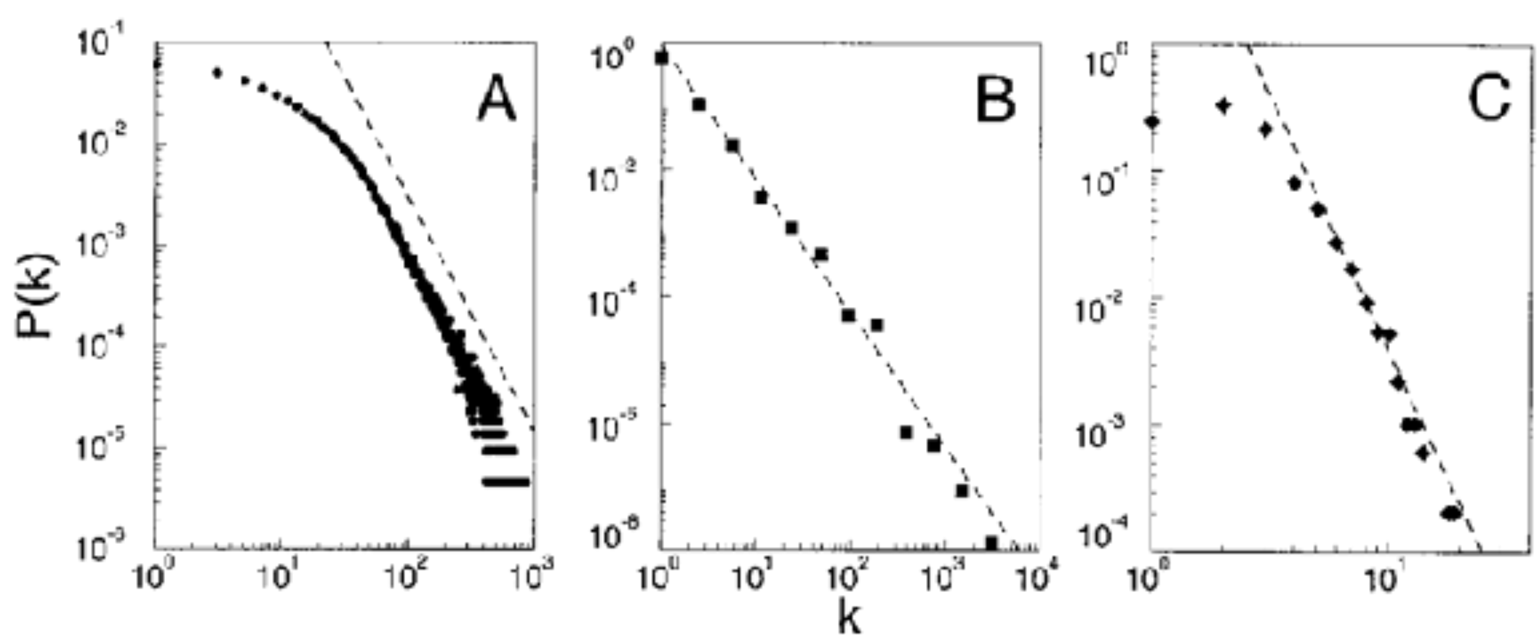
- Degree distribution: histogram of number of vertices of a certain degree





Heavy tail

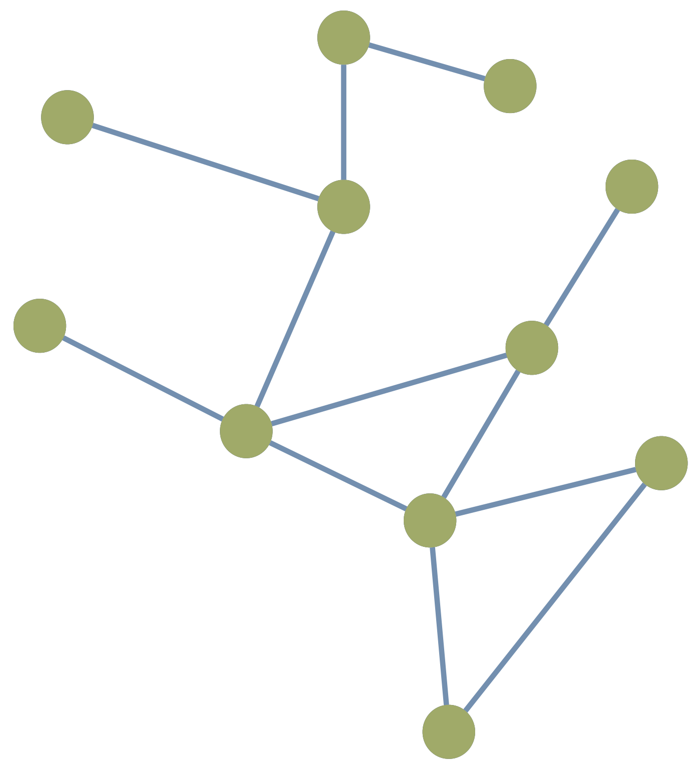


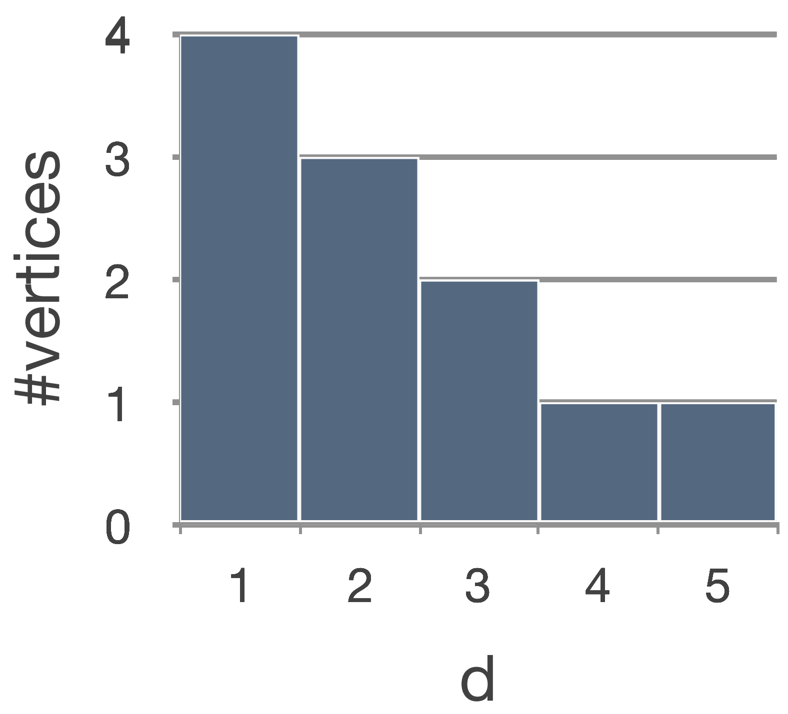


A: Actor collaboration network, B: WWW, C: Power Grid data [Barabási et. al., 1999]
Source: www.sciencemag.com

Why sample

- If access to whole graph: O(n) algorithm



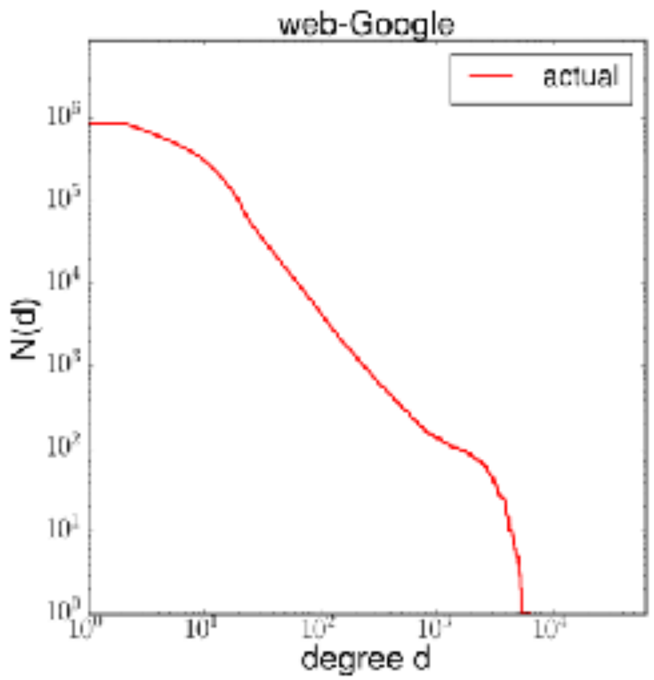


Why sample

- But what if we did not have access to whole graph?
  - Internet, routing networks
  - Crawl based methods, traceroutes [Faloutsos et. al., 1999]
    - Contains bias! [Achlioptas et. al., 2009]
    - Cannot simply scale sample.
  - [Faloutsos et. al., 1999], [Leskovec et. al., 2006], [Ebbes et. al., 2008] [Maiya et. al., 2011], [Ahmed et. al., 2010, 2014] - aim to capture representative graph sample

Problem Definition

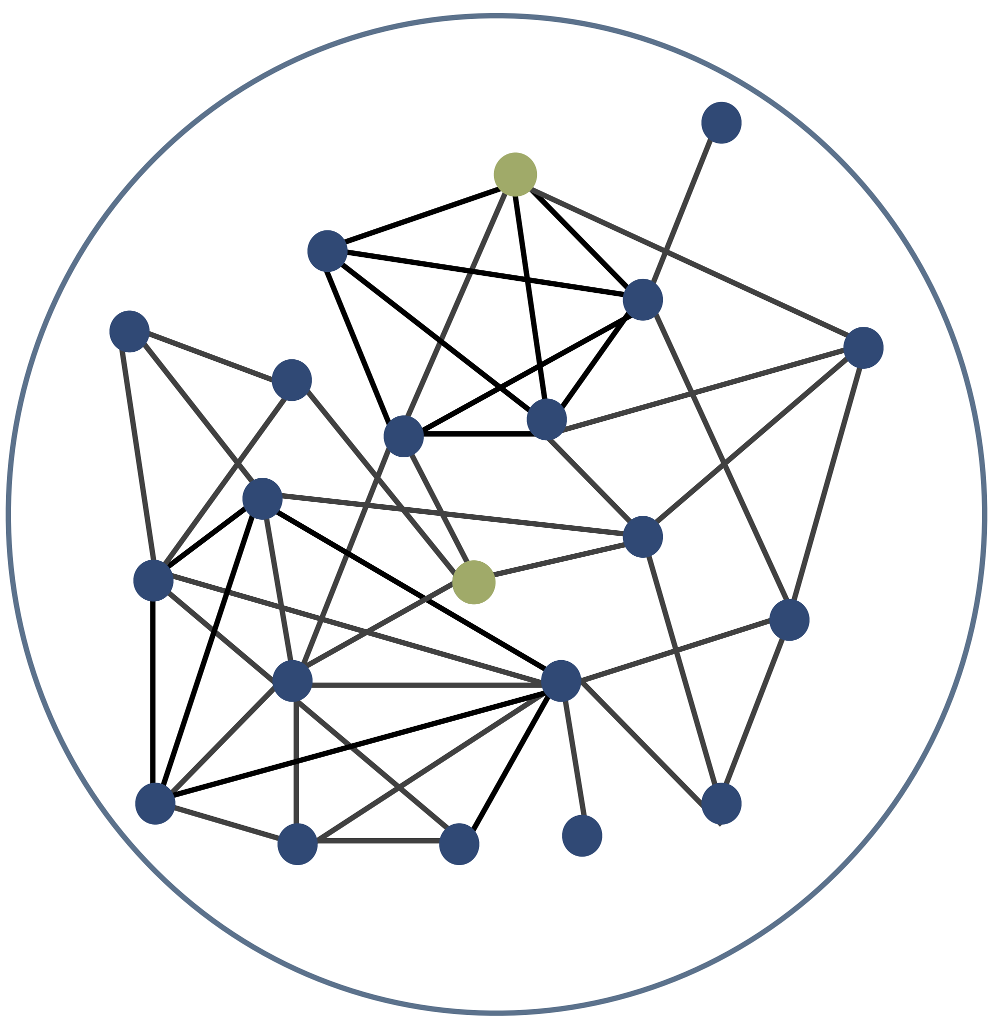
- ccdh: complementary cumulative degree histogram
  - N(d) = #vertices with degree **>=** d
- monotonically non-increasing, smooth



Can we estimate N(d) for any given d?

Query Model

1. Vertex queries: u.a.r. v ∈ V

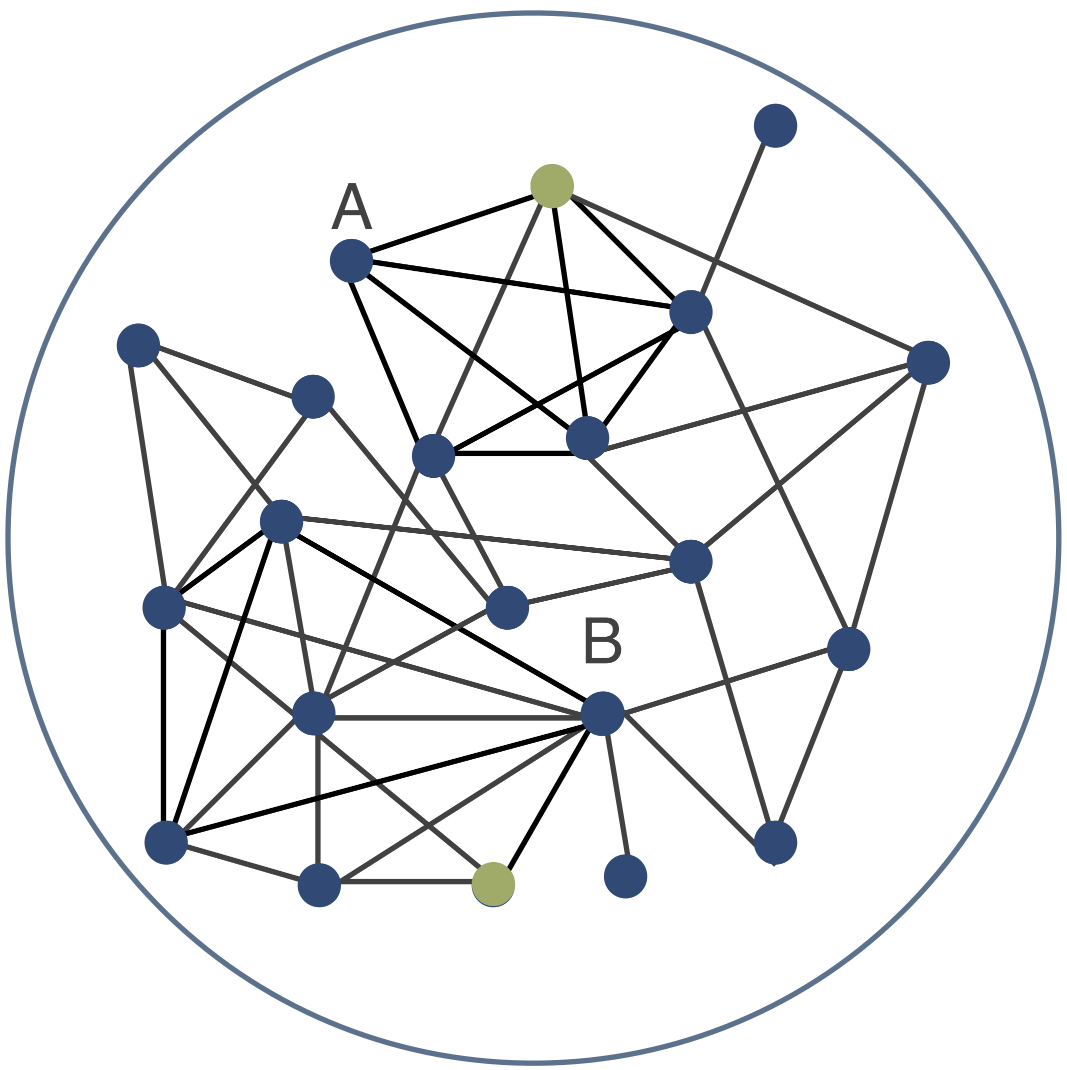


Can I get a vertex

Here you go!

Query Model

2. Neighbor queries: u.a.r. neighbor u of v

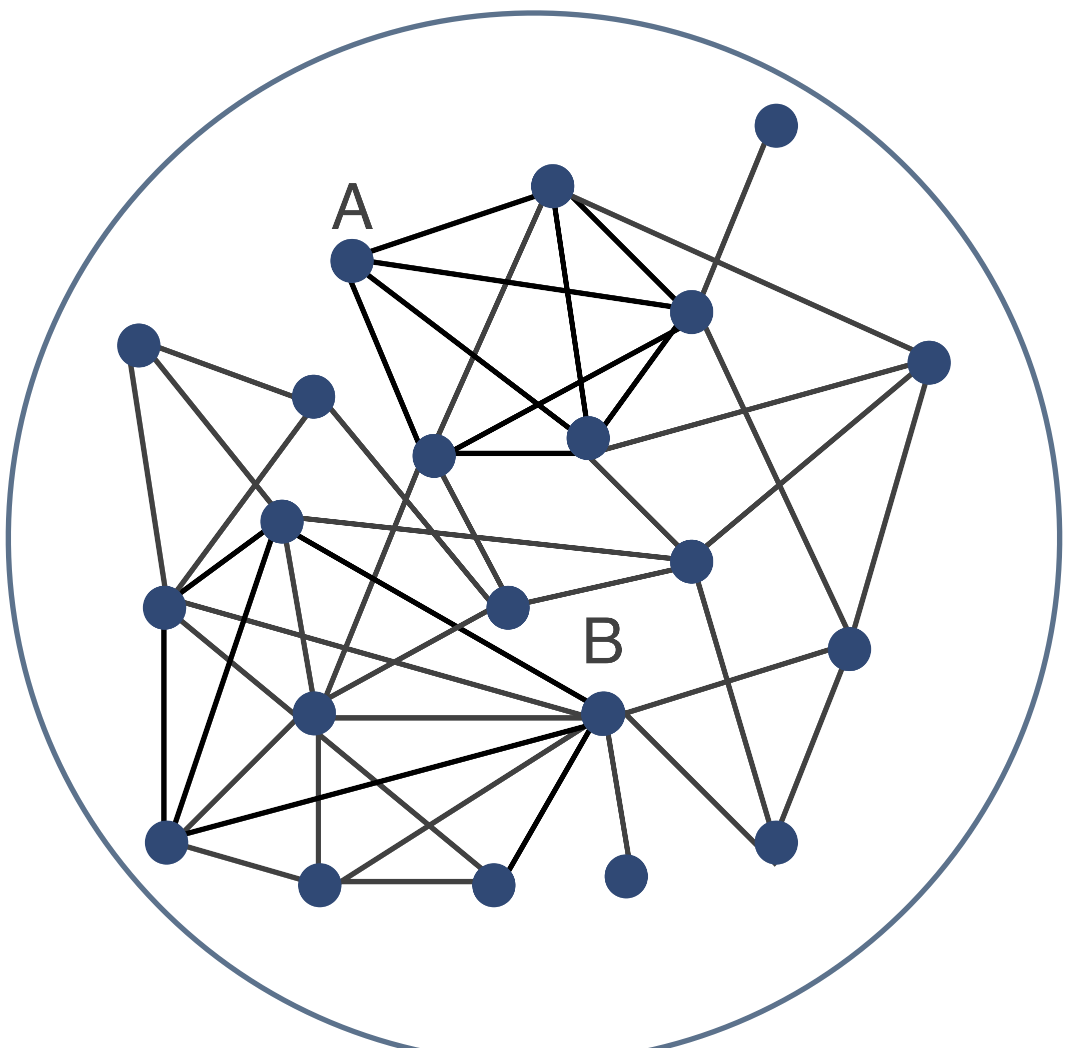


Can I have
a neighbor of AB

Here you go!

Query Model

3. Degree queries: degree d_v



Can I have
the degree of A

4

Query Model

1. Vertex queries: u.a.r. $v \in V$
2. Neighbor queries: u.a.r. neighbor u of v
3. Degree queries: degree d_v

Prior work

- Vertex sampling [Stumpf et. al., 2005, Lee et. al. 2006]
- Edge Sampling [Stumpf et. al., 2005, Lee et. al. 2006]

All need to sample at least 10-30% of the graph!

- Snowball Sampling [Maiya et. al., 2011]
- Linear system solver [Zhang et. al., 2015]

Main contribution

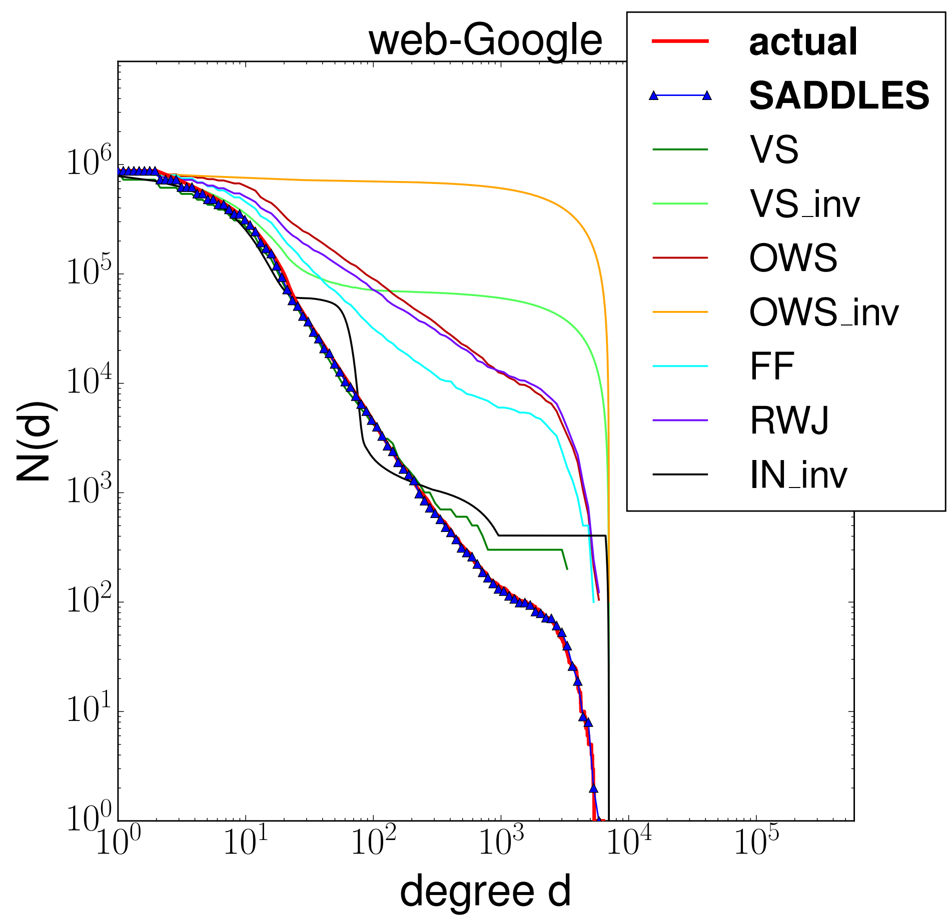
- Randomized algorithm SADDLES that estimates N(d)
- Uses a sublinear number of queries for any degree distribution bounded below by a power law.
- Power Law

exponent	number of samples
2	$n^{\frac{1}{2}}$
3	$n^{\frac{2}{3}}$

- Strongly sublinear!

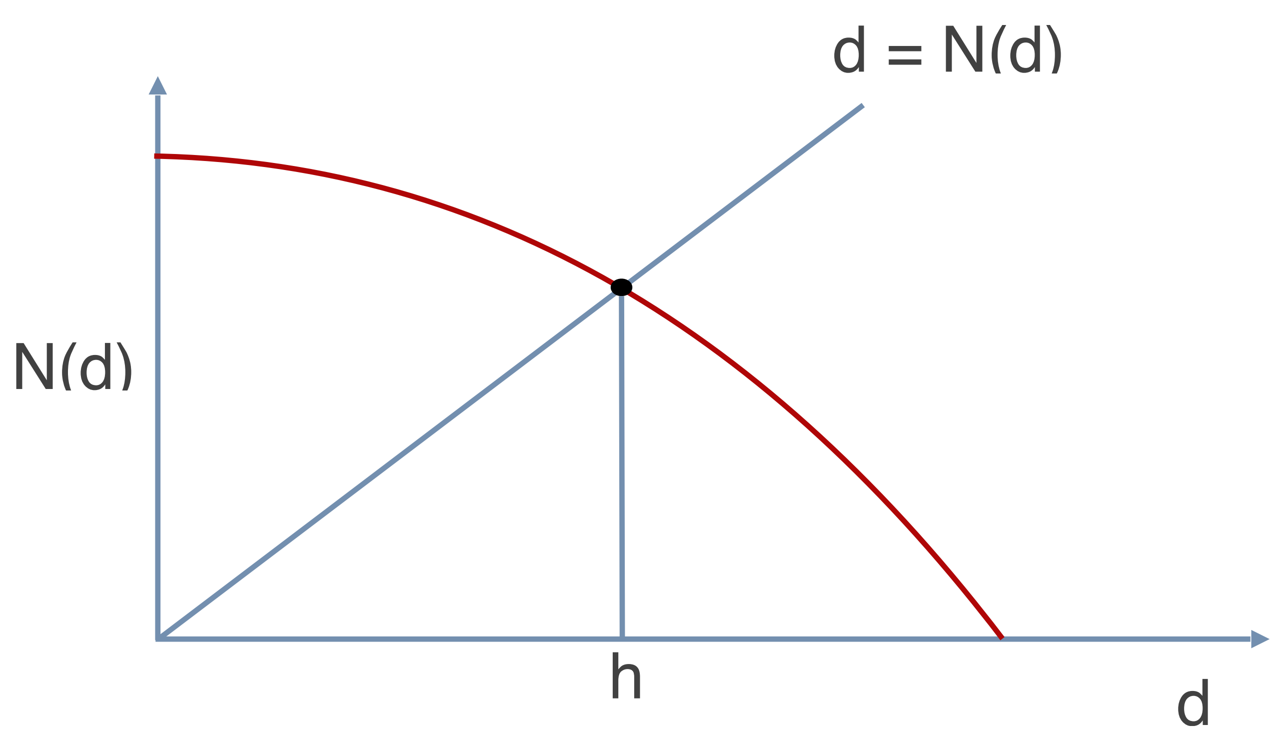
Main contribution

- In practice, we needed to sample only 1% of the graph
- Works well for all degrees



Query complexity

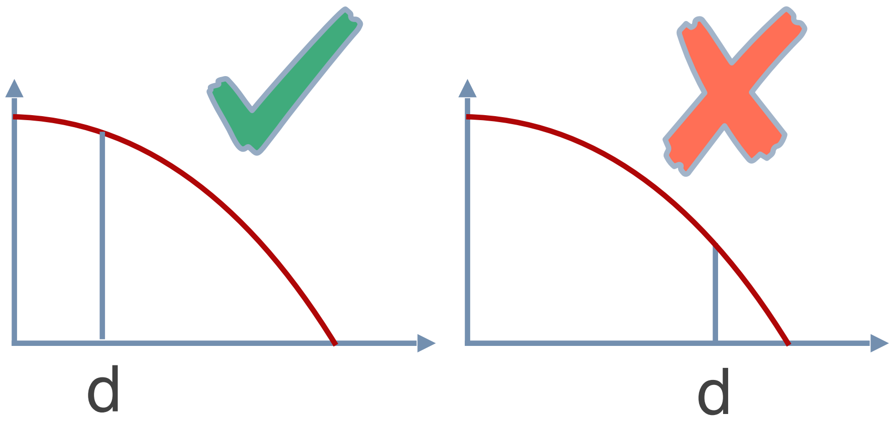
- Depends on 2 parameters:
  - h-index = $\min_d \max(d, N(d))$
    - Largest d, such that there are at least d vertices of degree >= d.
    - Same as the bibliometric h-index!



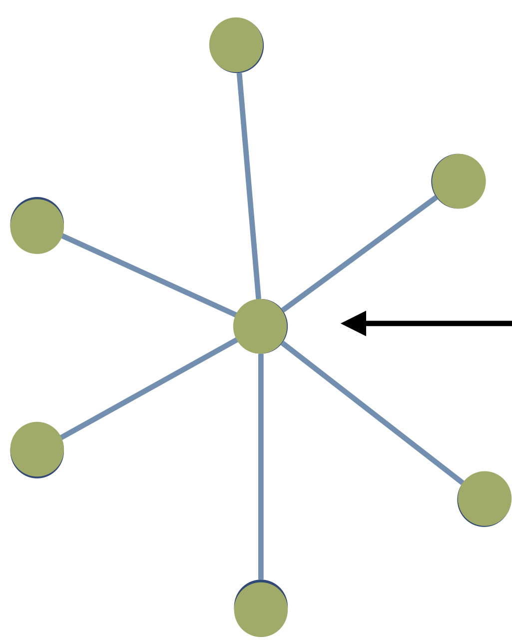
Query complexity

- Depends on 2 parameters:
  - h-index = $\min_d \max(d, N(d))$
  - z-index = $\min_{d:N(d)>0} \text{sqrt}(d \cdot N(d))$
    - replace max by geometric mean
- h and z are large for power laws!

Vertex sampling



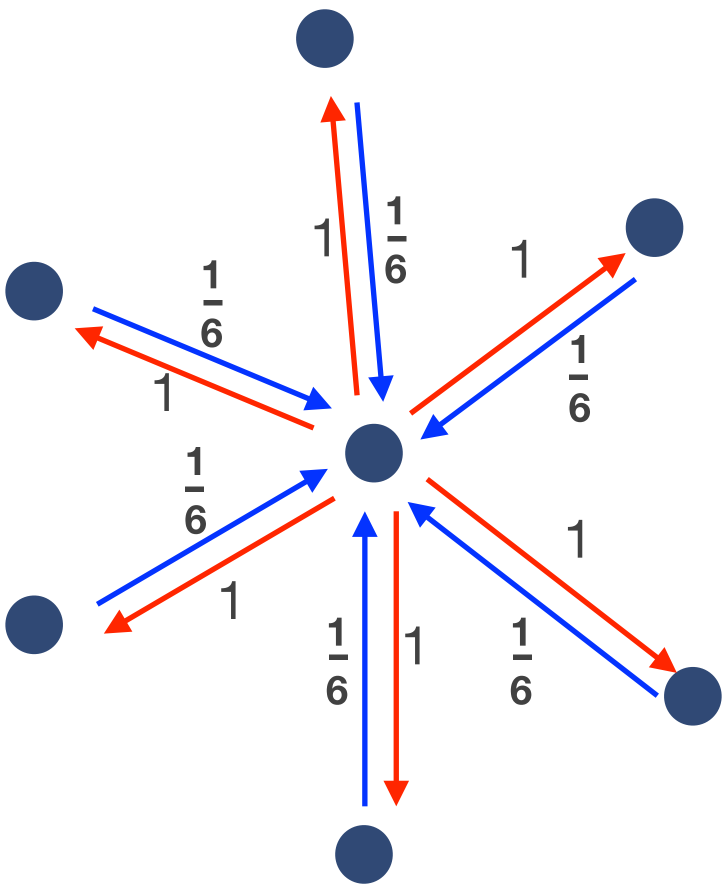
- Sample u.a.r. vertices
- Bin them according to degree
- Need $\dfrac{n}{N(d)}$ samples



Have to take many samples to hit high degree vertex

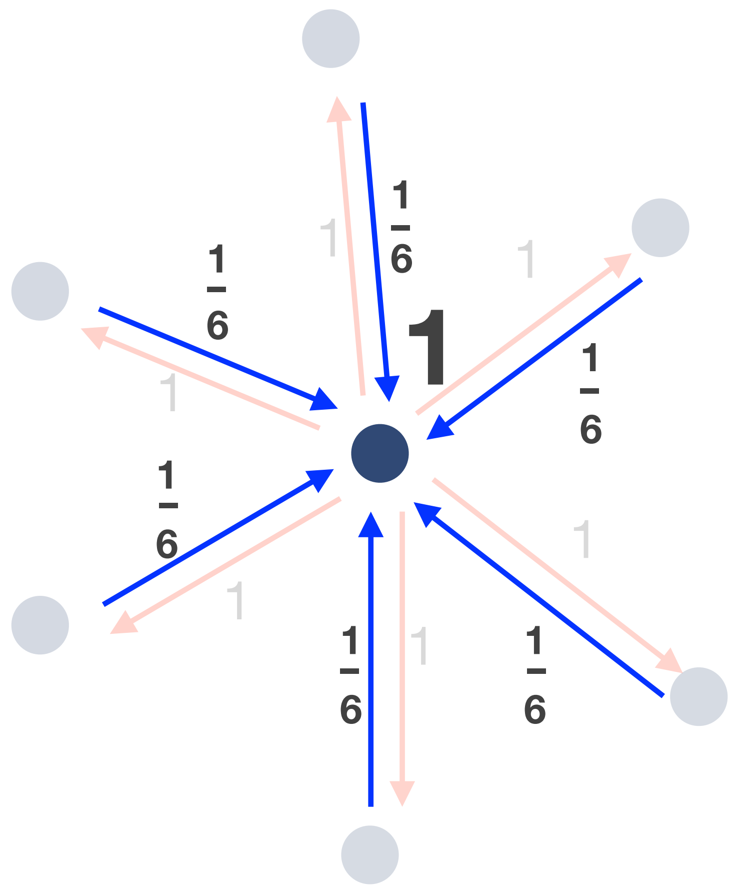
Edge sampling

Undirected edge -> 2 directed edges



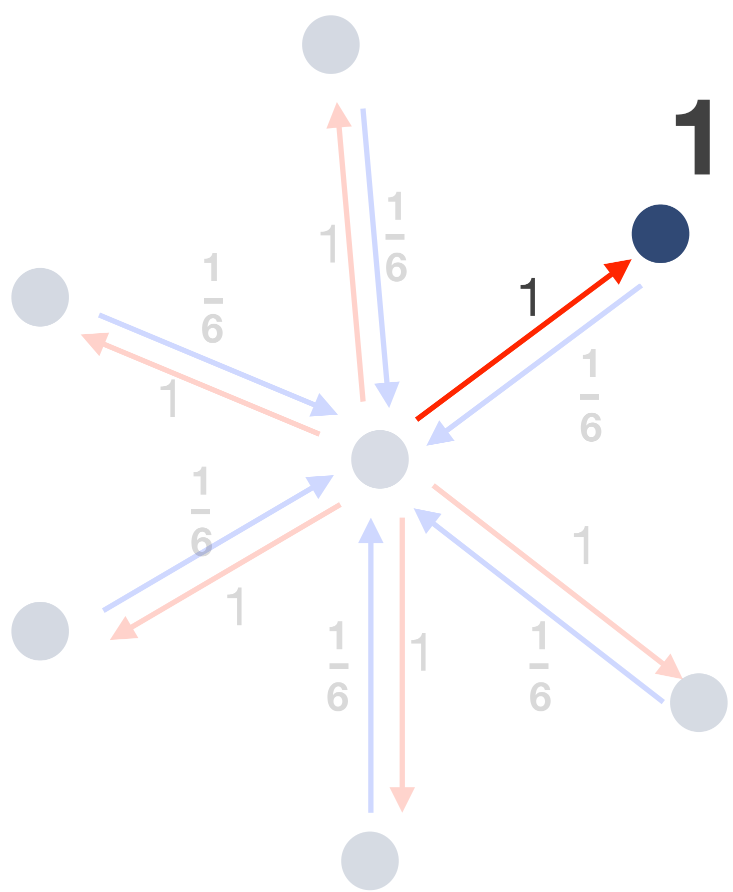
$$\mathrm{wt}((v,u)) = \frac{1}{d_u}$$

Edge sampling



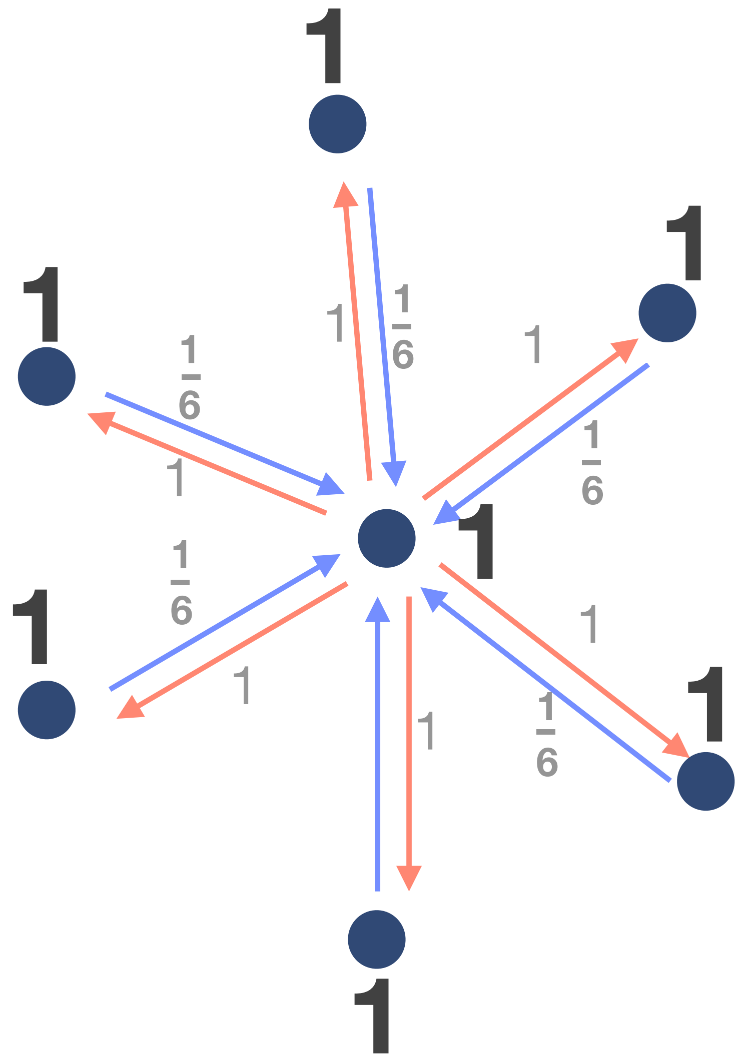
Sum of weights of edges incident on a vertex = 1

Edge sampling



Sum of weights of edges incident on a vertex = 1

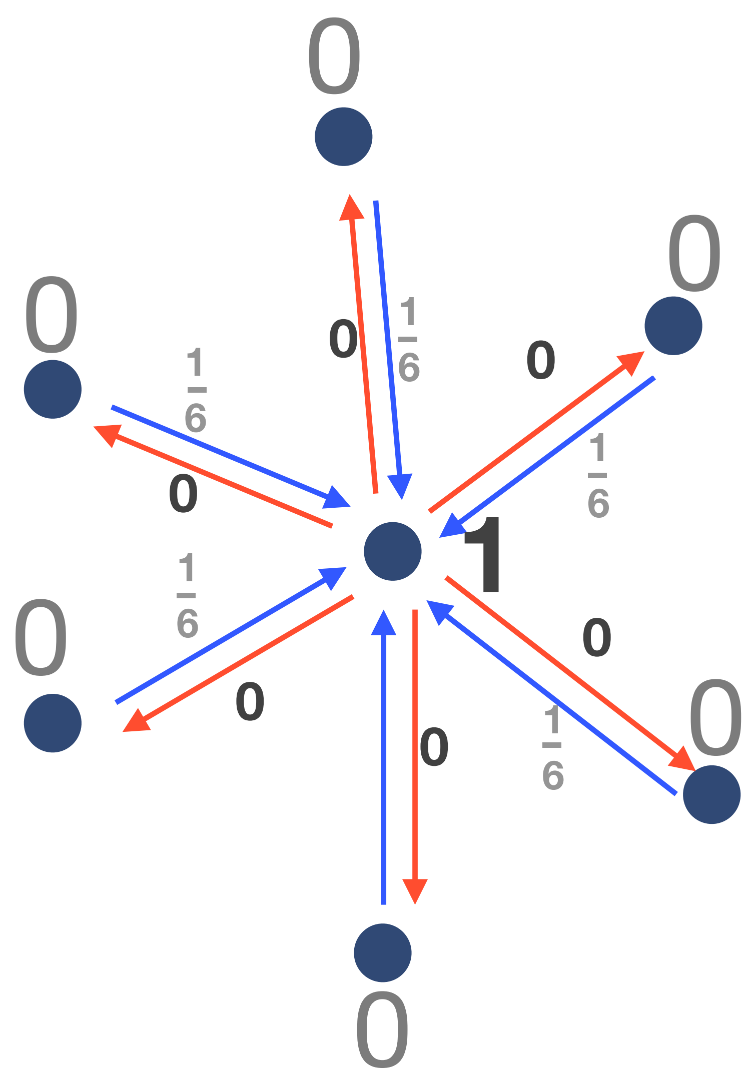
Edge sampling



Sum of all weights = n

Edge sampling

Say, d = 5

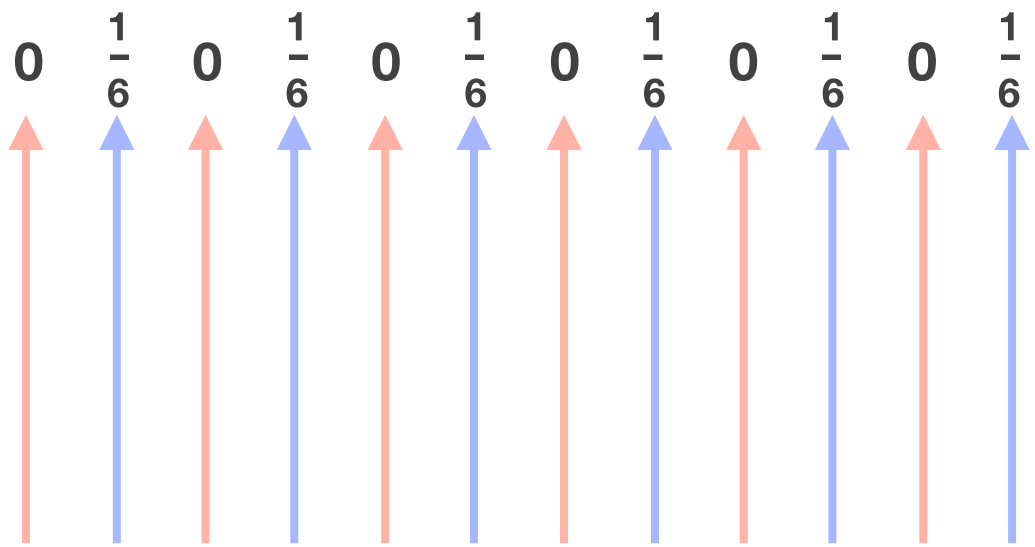


To get N(d), set weights of irrelevant edges to 0

Sum of all weights = N(d)

Edge sampling

$\boxed{\frac{2/6}{4}} \times 12 = 1$



Set of objects,
we want their sum

Sample randomly

Take average of
sampled weights

Scale by number of edges
to get total sum

Main Idea

- Combine vertex sampling and edge sampling
- But we don't have edge sampling
- Simulate it!

Theoretical work

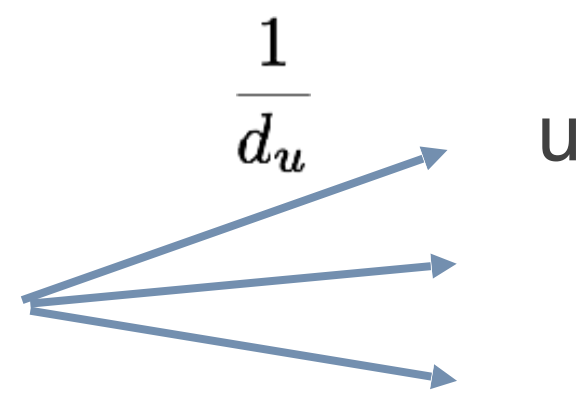
- Average degree [Feige et. al., 2006], [Goldreich et. al., 2002, 2008]
- Number of star graphs, moments [Eden et. al., 2011]
- Number of triangles [Eden et. al., 2014]

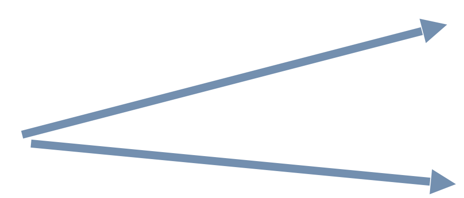
Simulated Edge Sampling

- Sample some vertices
- The neighbors of these vertices is the edge set that we will perform random sampling on.

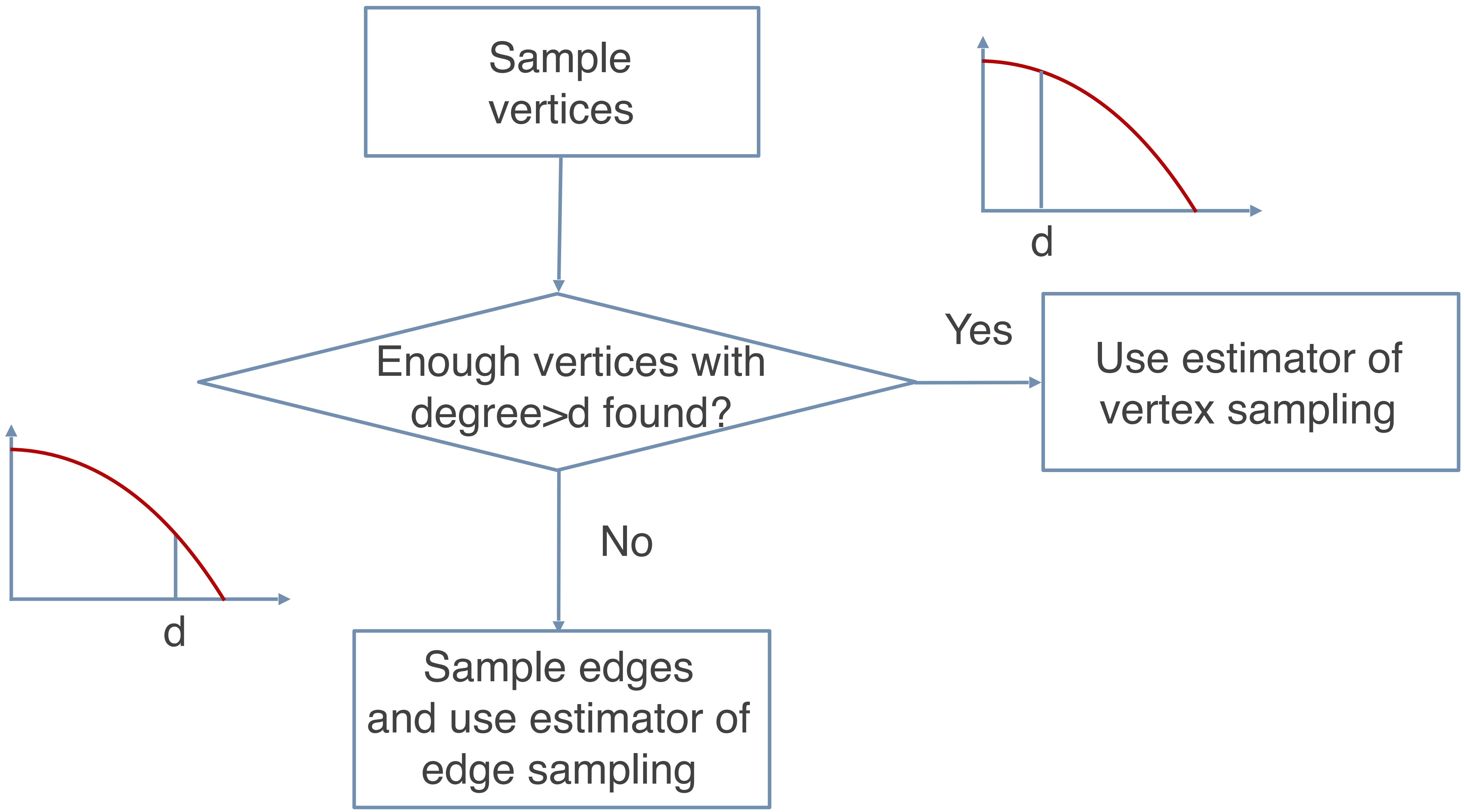
Simulated Edge Sampling

- Sample r vertices
- Set up distribution D to sample vertex v $\propto$ d_v
- Repeat q times:
  - Sample a vertex v from D
  - Sample u.a.r. neighbor u of v
- Find average weight of samples
- Scale appropriately





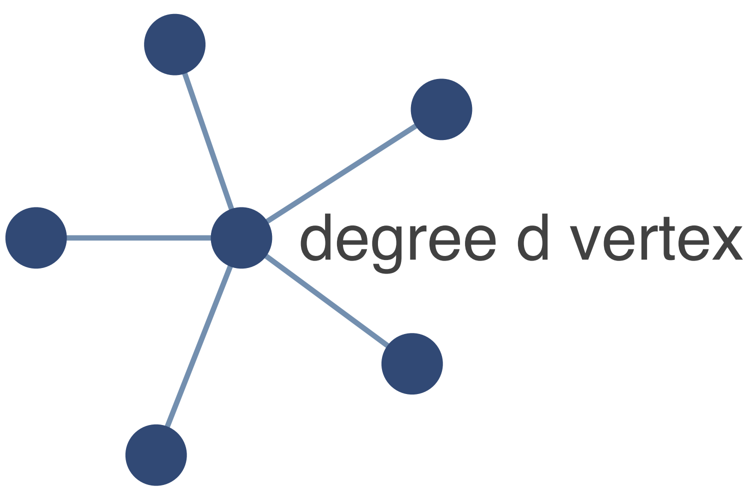
Putting it all together



r and q

- Total samples: $O^*(r+q)$
- How big do r and q need to be?
  - If VS: r = $O^*(\frac{n}{N(d)})$
  - If ES: r = $O^*(\frac{n}{d})$
  - Similarly,

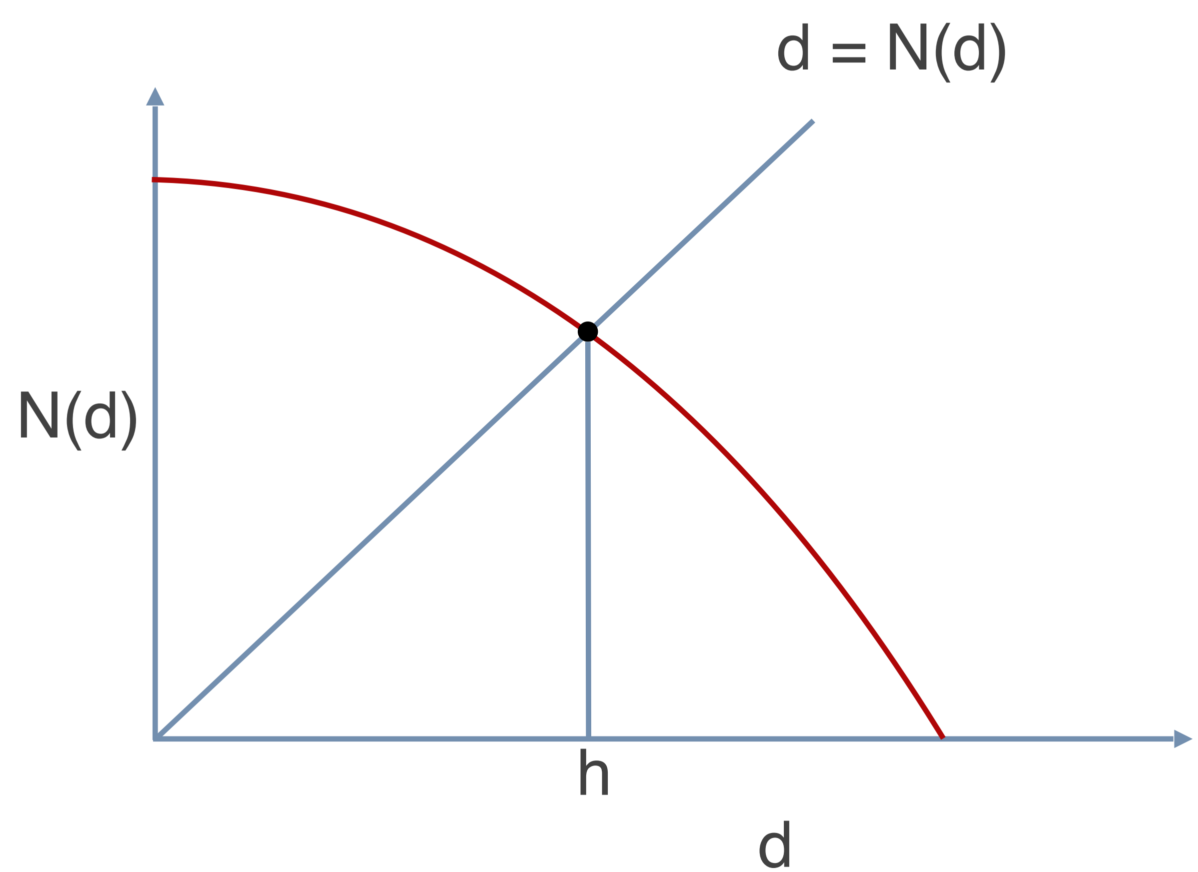
    q = $O^*(\frac{m}{dN(d)})$



Want at least 1 of its
d neighbors to be in R

Query complexity

- Query complexity: $O^*(\frac{n}{h} + \frac{m}{z^2})$
- Vertex queries: $O^*(\frac{n}{max(d,N(d))})$
- Neighbor queries: $O^*(\frac{m}{dN(d)})$



Simulated Edge Sampling

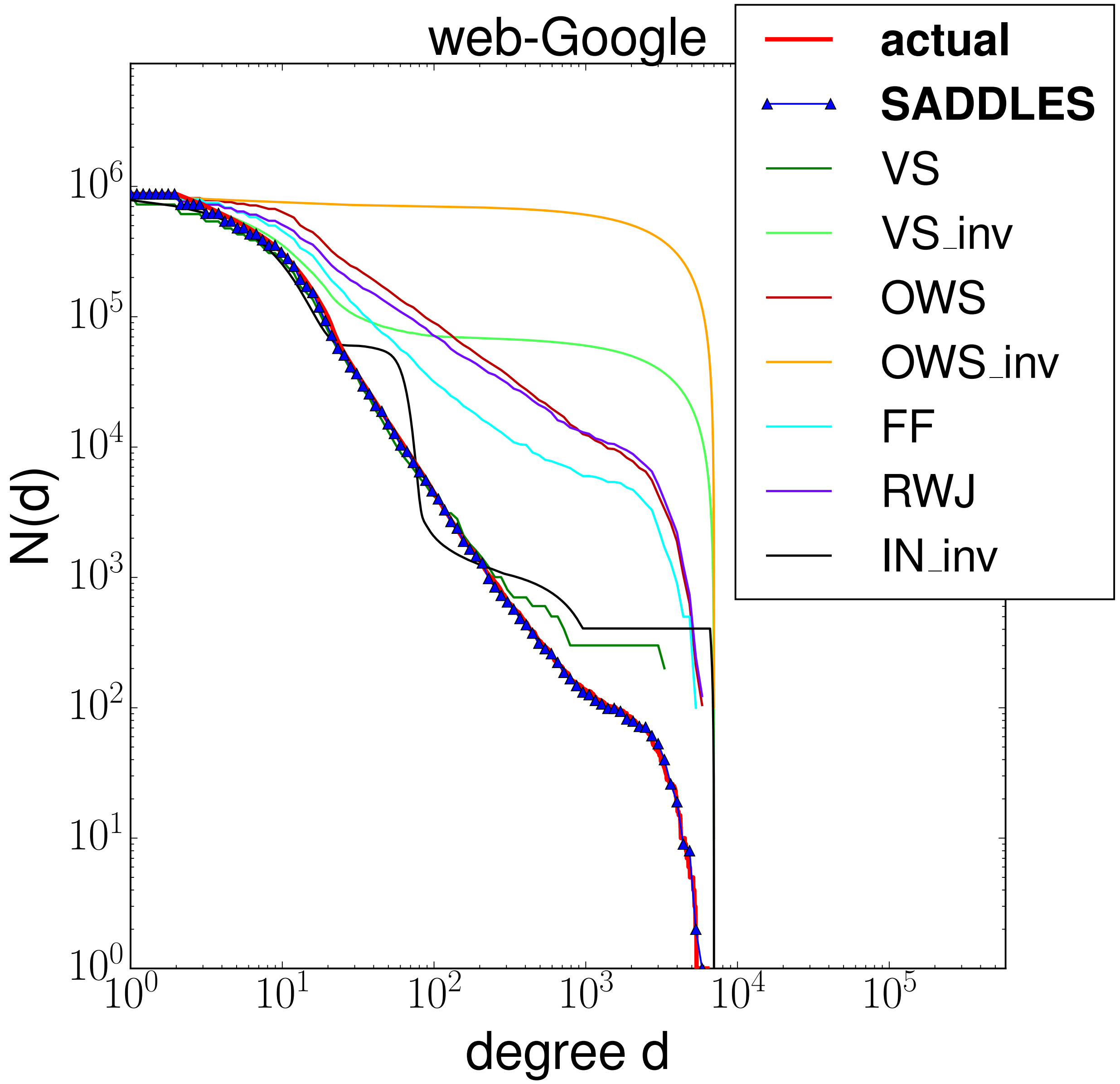
- Single edge sample is uniform at random
- But multiple edge samples are correlated
- Key insights:
  - Correlation can be contained if h and z are high. Power laws have high h and z!
  - 1-hop distance is enough - don’t need to do long random walks

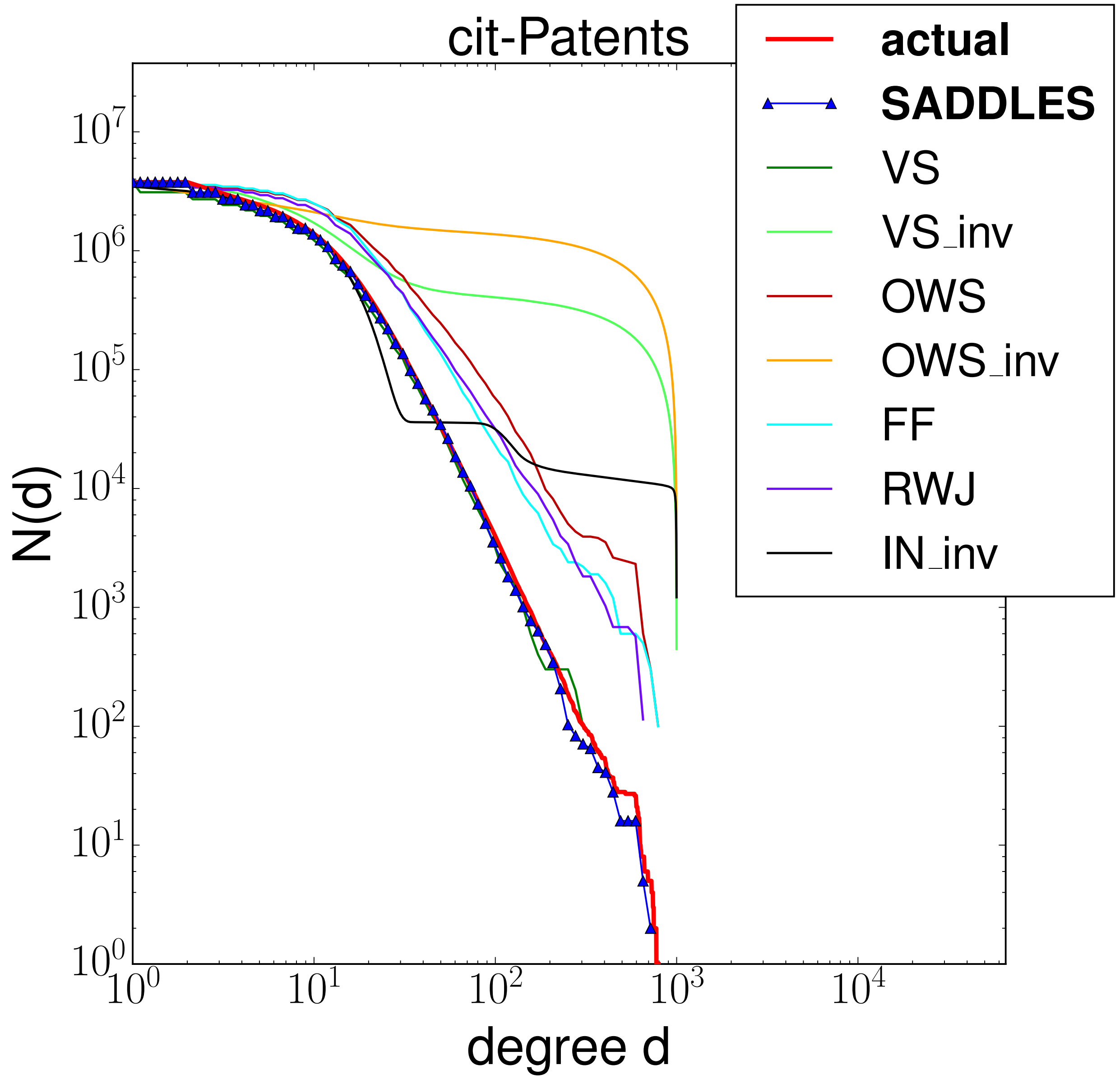
h and z

- Indeed large!

GRAPH	VERTICES	EDGES	AVG. DEG.	h	z
web-BerkStan	0.6M	6M	10	**707**	**220**
as-skitter	2M	11M	7	**982**	**184**
com-lj	4M	34M	9	**810**	**114**
com-orkut	3M	110M	38	**1638**	**172**

Results


web-Google
actual
SADDLES
VS
VS_inv
OWS
OWS_inv
FF
RWJ
IN_inv
N(d)
degree d
10^0
10^1
10^2
10^3
10^4
10^5
10^6


cit-Patents
actual
SADDLES
VS
VS_inv
OWS
OWS_inv
FF
RWJ
IN_inv
N(d)
degree d
10^0
10^1
10^2
10^3
10^4
10^5
10^6
10^7

Thank you

Questions?