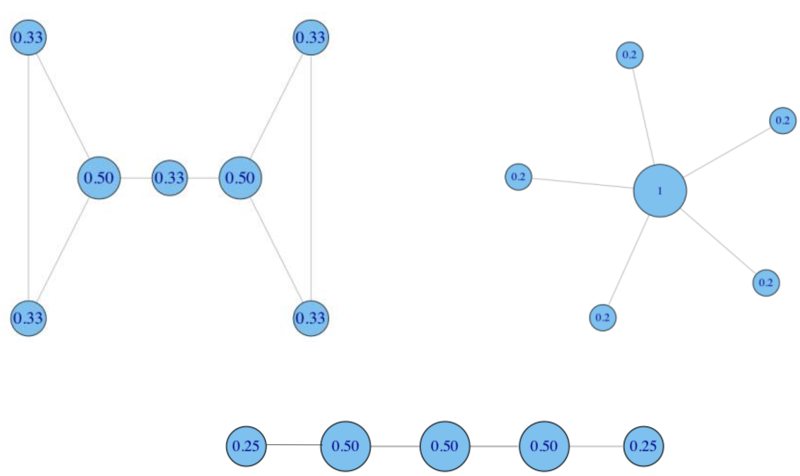
Graph Centrality

- Degree centrality
  - The node degree
- Closeness centrality
  - (the sum of distances to all other nodes)$^{-1}$
- Betweenness centrality
  - The number of shortest path thru a node
- Eigenvector centrality

degree: normalized degree centrality

divide by the max. possible, i.e. (N-1)



35

betweenness centrality: definition

betweenness of vertex i

paths between j and k that pass through i

$$C_B(i) = \sum_{j<k} g_{jk}(i) / g_{jk}$$

all paths between j and k

Where g_{jk} = the number of geodesics connecting *j*-*k*, and
$g_{jk}(i)$ = the number that node *i* is on.

Usually normalized by:

$$C_B^{'}(i) = C_B(i) / \boxed{[(n-1)(n-2)/2]}$$

number of pairs of vertices excluding the vertex itself

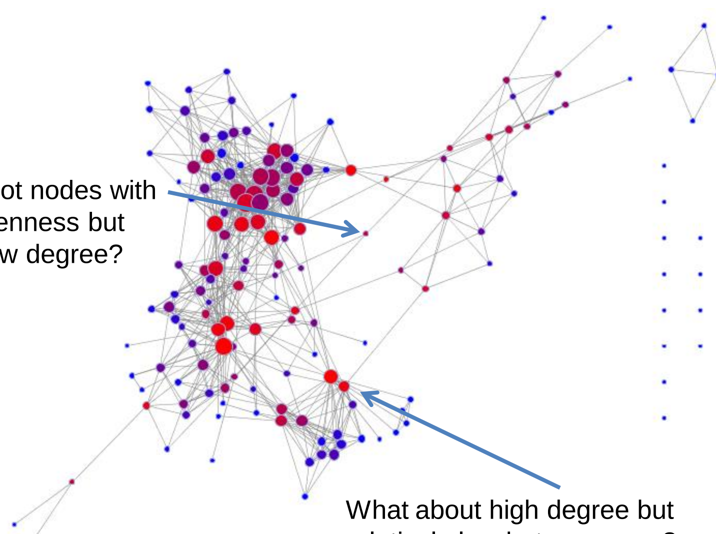
directed graph: (N-1)*(N-2)

36

example

Nodes are sized by degree, and colored by betweenness.

Can you spot nodes with high betweenness but relatively low degree?



What about high degree but relatively low betweenness?

37

Eigenvector Centrality

Adjacency matrix

$$x_i = \frac{1}{\lambda} \sum_{j \in N_i} x_j = \frac{1}{\lambda} \sum_{j=1}^{N} A_{ij}\, x_j$$

$\boxed{\lambda X = AX}$

X is the vector of Eigenvalues

PageRank is conceptualy similar

The Graph Diameter

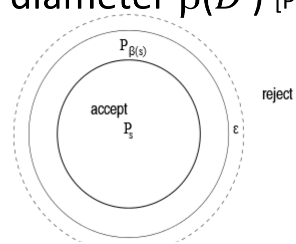
- Diameter = the maximal distance between all pairs of vertices

Unweighted graph

- Shortest path from a node to all others: $\Theta(m)$
- All pair shortest path: $\Theta(n \cdot m)$ (using BFS)
- Using matrix product: $O(n^{2.376} \text{ polylog}(n))$
  - And $\Theta(n^2)$ space [Alon *et al*., FOCS 1992]
- Fast algorithms that use $\Theta(n^2)$ space
  - $\Theta(n^3/\log n)$ for dense graphs [Feder & Motwani, STOC 91]
  - $O(n^2(\log \log n)^2/\log n)$ for sparse graphs [Chan, SODA'06]

Unweighted Graph (cont.)

- Estimating D by $\overline{D}$ [Dor *et al.,* 1997]
  - $\overline{D} \le D \le \overline{D} + 2$
  - Time $\Omega(n^2)$
  - Space $\Theta(n^2)$
- Testing if the diameter is below $\overline{D}$ or the graph is ε-far from a graph with diameter $\beta(\overline{D})$ [Parnas & Ron,2002]



Bounds

- Trivial bounds:
  For any vertex *v*: ecc(*v*) ≤D≤2·ecc(*v*)
  - ecc(*v*) is the eccentricity of *v*.
  - Can be computed in Θ(m) time & space
- Double sweep lower bound:
  choose *v* s.t. d(*v,u*)=ecc(*u*) for some *u*.
  - for trees (and other special graphs) D=ecc(*v*)
  - For other graphs it is a tighter lower bound
  - Can be computed in Θ(m) time & space
- Tree upper bound:
  for any spanning tree, the tree diameter is an u.b.
  - Can be computed in Θ(m) time & space

Discussion

- Iterate for different vertices can improve the bounds
- For the tree upper bound, chose the highest degree node
  - Good for power-law graphs
- Iterating may not always help
  - E.g., the tree u.b. for a cycle

Experimnets

- Internet router graph from Skitter (2005)
  n=1,719,037 m=11,095,298
- A web graph (.uk domain, 2005)
  n=39,459,925 m=783,027,125
- Peer to peer graph (eDonkey sharing, 2004)
  n=5,792,297 m=142,038,401
- IP traffic graph
  n=2,250,498 m=19,394,216

Simulation Results

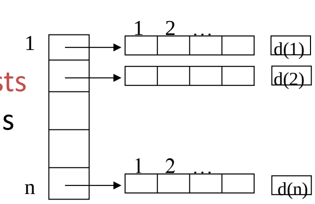
	t.l.b.		d.s.l.b.		h.d.t.u.b.		r.t.u.b.		t.u.b.		iterations
INET	29		**31**		**34**		34		38		10,000
	998	0.0001	2	0.49	26	0.1002	23	0.0633	127	0.0011	
P2P	8		**9**		**10**		10		10		5,000
	210	0.0094	1	0.7005	1	0.039	120	0.0032	3237	0.0001	
WEB	26		**32**		**33**		33		34		2,000
	816	0.001	1	0.985	34	0.0015	46	0.0025	1572	0.0005	
IP	9		**9**		**9**		9		10		10,000
	5331	0.0001	1	0.989	4	0.0543	12	0.0346	6284	0.0001	

Portion of iteration hitting the bound

First iteration to hit the bound

Testing the Graph Diameter

- We assume graphs are represented by the incidence lists of the vertices, where each list is accompanied by its length.
- Allowed queries:
  - what is the degree, d(v), of any vertex v?
  - who is the i'th neighbor of v, for any vertex v and index 1≤ i ≤ d(v)?



Parnas & Ron, 1992

ε-far Definition

Let P_s be a fixed parameterized property, $0<\varepsilon<1$, and m a positive integer. A graph G having at most m edges is ε-far from property P_s (with respect to the bound m), if the number of edges that have to be added and/or removed from G in order to obtain a graph having property P_s, is greater than $\varepsilon \cdot m$.

Otherwise, G is ε-close to P_s.

A Testing Algorithm

A testing algorithm for (parametrized) property P_s, with boundary function $\beta(\cdot)$, is given a (size) parameter $s>0$, a distance parameter $0<\varepsilon<1$, a bound $m>0$, and a query access to an unknown graph G having at most m edges.

The output of the algorithm is *accept* or *reject*.

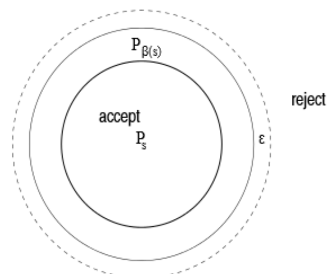
- If G has property P_s, then the algorithm should output accept with prob. at least 2/3
- If G is ε-far from property P_s, then the algorithm output reject with prob. at least 2/3.

Diameter Testing

Testing if a graph diameter is bounded by D.

A family of algorithms which differ in:

- The boundary function β(·)
- Query and time complexity
- The value of ε



Why Only small ε?

- Every connected graph with *n* vertices can be transformed into a graph with diameter at most *D* by adding at most $n/\lfloor D/2 \rfloor$ edges.

$\Downarrow$

- Every connected graph with *n* vertices and *m* edges is ε-close to having diameter *D* for every $\varepsilon \geq \frac{2}{D} \cdot \frac{n}{m}$

$\Downarrow$

- $\varepsilon_{n,m} \stackrel{\text{def}}{=} \frac{m}{n} \cdot \varepsilon$

Main Results
Testing algorithms for diameter *D*

1. Boundary function $\beta(D) = 4D+2$
   Query time $O(1/\varepsilon_{n,m}{}^3)$
   1-sided error: always accept graphs with diameter at most *D*.
2. Boundary function $\beta(D) = 2D+2$
   Query time $O(\frac{1}{\varepsilon_{n,m}{}^3} \cdot \log^2 \frac{1}{\varepsilon_{n,m}})$
   2-sided error
3. Boundary function $\beta(D) = D(1+\frac{1}{2^i-1})+2$, $2 \le i \le \log(D/2+1)$
   Query time $O(\frac{1}{\varepsilon_{n,m}{}^3} \cdot \log^2 \frac{1}{\varepsilon_{n,m}})$, $\varepsilon = \Omega(\frac{n^{1-\frac{1}{i+2}} \cdot \log n}{(i+2)m})$

$$\varepsilon_{n,m} = \Omega(\frac{n^{-\frac{1}{i+2}} \cdot \log n}{(i+2)})$$

Main Results

	$\beta(D)$	ε	Remarks
1.	2D+2	Any	One Sided Error
2.	$\left(1+\frac{1}{2^i-1}\right) \cdot D+2$	$\Omega\left(\frac{n^{1-\frac{1}{i+2}} \cdot \log n}{(i+2) \cdot m}\right)$	Two Sided Error
	4D/3 + 2	$\tilde{\Omega}\left(n^{-1/4}\right)$	i = 2
	D+4	$\Omega(1/\mathrm{poly}(\log n))$ for D = poly(logn)	i = log(D/2 + 1)

Time and Query Complexity: $\tilde{O}\left(\frac{1}{\varepsilon^3}\right)$

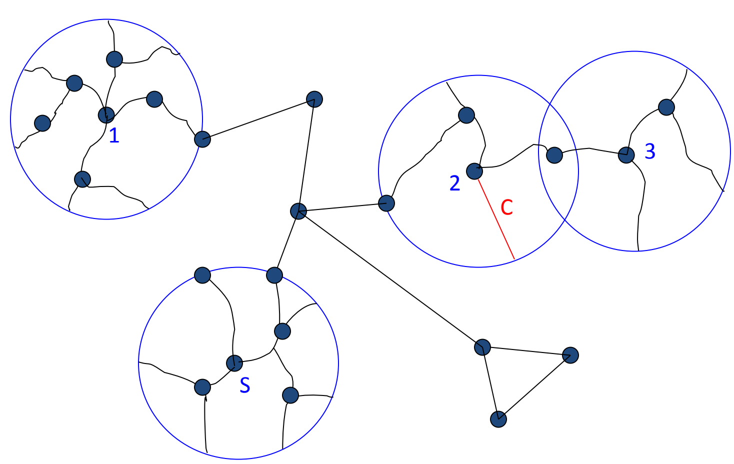
Algorithm

Input: D, n, m, ε.
Parameters: C, k, α.

- Set $\varepsilon_{n,m} = \frac{m}{n} \cdot \varepsilon$
- Uniformly select $S = \Theta(1/\varepsilon_{n,m})$ starting vertices.
- For each starting vertex - perform a BFS to distance at most C until k vertices are reached.
- If at most $\alpha \cdot S$ starting vertices reach < k vertices then accept, otherwise reject.

Time and Query Complexity: $O(k^2 \cdot S) = O(k^2/\varepsilon_{n,m})$

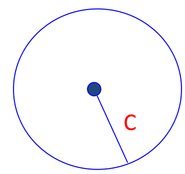
Illustration of the Algorithm



Proof of Correctness

Good Vertex: If C-neighborhood contains ≥ k vertices.

Bad Vertex: If C-neighborhood contains < k vertices.



We Show:

- Diameter ≤ D ⟹ Almost (all) vertices are good.
- Diameter > β(D) ⟹ Many vertices are bad.

Reducing the Diameter

Lemma 1:
If at least (1-1/k)n of the vertices are good, then the graph can be transformed into a graph with diameter at most 4C+2 by adding at most 2n/k edges.

Proof: