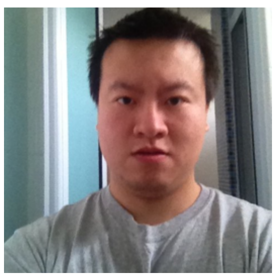
Make It or Break It: Manipulating Robustness in Large Networks

Hau Chan
Stony Brook University



Leman Akoglu
Stony Brook University

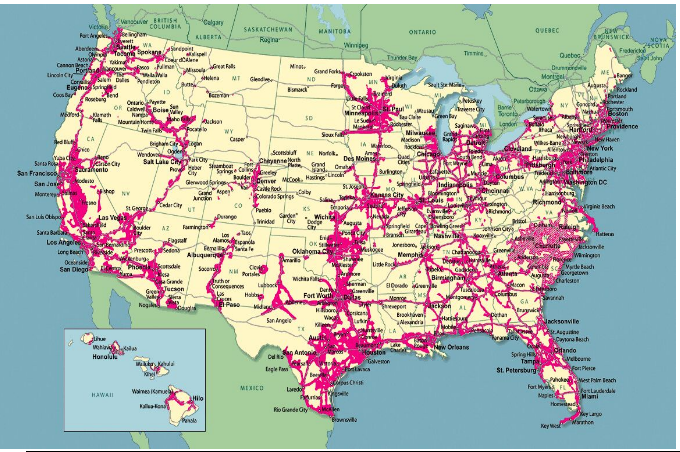
Hanghang Tong
City College of New York

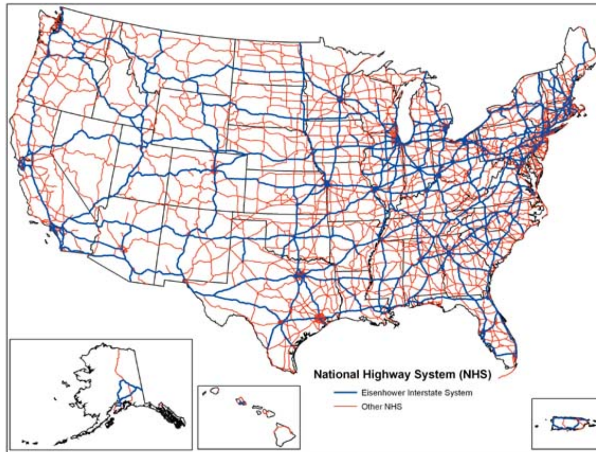
SIAM SDM
Philadelphia, PA
April 24-26, 2014

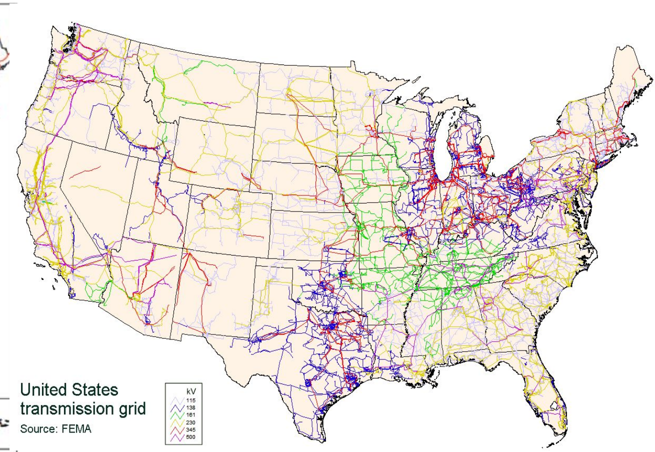
Networked systems are everywhere!

Stony Brook University Computer Science

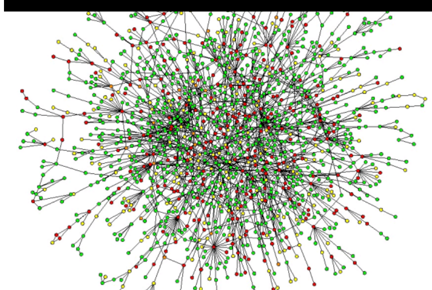
CUNY



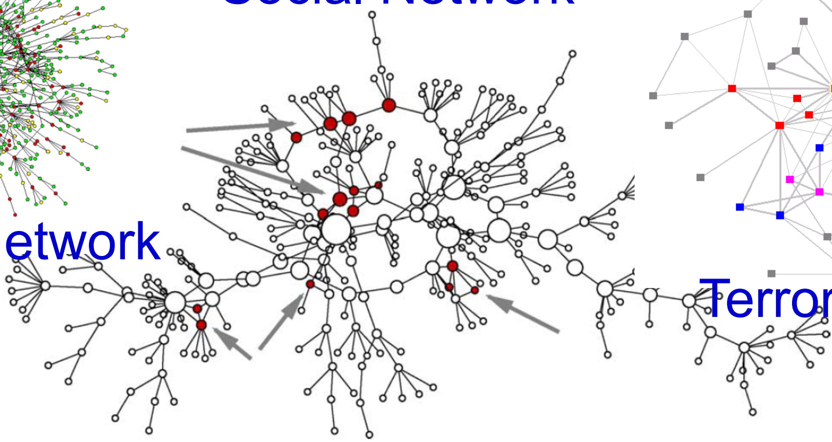




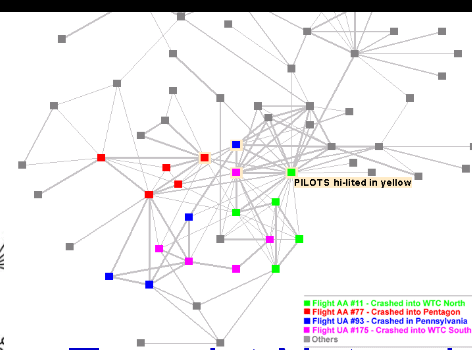
How robust are these networks?



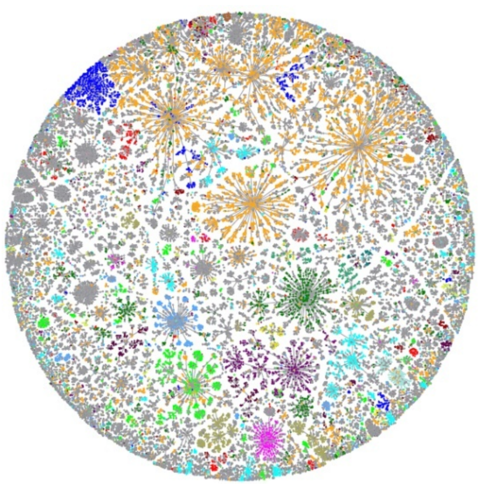
Biological Network



Social Network



Terrorist Network



Network Robustness

- Robustness is the ability of a network to continue performing well when it is subject to failures or attacks.
  - random failure (server down)
  - cascading failure (virus propagating)
  - targeted attack (carefully-chosen agents down)
- Goal: precise definition that can be computed
  - Computable measure allows to:
    - compare two networks
    - modify existing network to improve its robustness
    - design robust new network

Main questions

Q1 How to measure the robustness of a given network?

Q2 How to modify a given network to improve its robustness?

Main questions

Q1 How to measure the robustness of a given network?

- interpretable
- (strictly) monotonic
- captures redundancy
- …

Q2 How to modify a given network to improve its robustness?

Main questions

Q1 How to measure the robustness of a given network?

- interpretability, monotonicity, redundancy, …

Q2 How to modify a given network to improve its robustness?

- under policies: node/edge deletion, edge rewiring, edge addition, …
- subject to constraints: cost, #agents to modify, connectivity constraints between agents, …

Today's Roadmap

- ✓ Network Robustness
  - Intro
  - Main Questions
- ➡ Measuring Robustness **Q1**
- Modifying Robustness **Q2**

Robustness Measures

- Study of robustness:
  - mathematics, physics, computer science, biology
- A long (!) and profoundly diverse list of measures:
  - vertex/edge connectivity
  - avg. shortest distance
  - max. shortest distance (diameter)
  - efficiency
  - vertex/edge betweenness
  - clustering coefficient
  - largest component fraction/avg. component size
  - total pairwise connectivity
  - average available flows

Robustness Measures

- …
- algebraic connectivity
- effective resistance
- number of spanning trees
- principal eigenvalue λ_1
- spectral gap $\lambda_1 - \lambda_2$
- natural connectivity
- other (combinatorial) measures:
  - toughness, scattering number, tenacity, integrity, fault diameter, isoperimetric number, min balanced cut, restricted connectivity, …

(algebraic connectivity, effective resistance, number of spanning trees:) eigenvalues of the Laplacian **L**

(principal eigenvalue, spectral gap, natural connectivity:) eigenvalues of the adjacency **A**

Robustness Measures

Stony Brook University Computer Science
CUNY

- …
- algebraic connectivity
- effective resistance
- number of spanning trees

} eigenvalues of the Laplacian **L**

- principal eigenvalue λ_1
- spectral gap $\lambda_1 - \lambda_2$
- natural connectivity

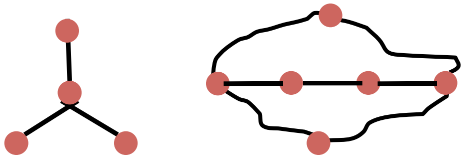
} eigenvalues of the adjacency **A**

- other (combinatorial) measures:
  - toughness, scattering number, tenacity, integrity, fault diameter, isoperimetric number, min balanced cut, restricted connectivity, …

an avalanche (!) of measures…
which one(s) to use?

A “guide” for “good” measures

- Strict monotonicity
  - improves strictly when edges are added
  - *related: differentiates graphs



- Redundancy
  - accounts for alternative/back-up paths
- Stability
  - does not change drastically by small changes
  - *related: meaningful for disconnected graphs
- Interpretability
  - its meaning is intuitively clear

A “guide” for “good” measures

Measures	S. Monotone	Redundant	Stable	Interpretable
vertex / edge connectivity	✗		✗	✓
avg. shortest distance	✗	✗	✗	✓
diameter	✗	✗	✗	✓
efficiency	✓	✗	✓	✓
vertex / edge betweenness	✓	✗	✗	✓
clustering coefficient	✗		✓	✓
largest component fraction	✗	✗		✓
total pairwise connectivity	✗	✗		✓
avg. available flows		✓	✗	✓
algebraic connectivity	✗		✗	✗
effective resistance	✓	✓	✓	✓
number of spanning trees	✗		✗	
spectral radius / gap			✓	✗
natural connectivity	✓	✓	✓	✓

A “guide” for “good” measures

Measures	S. Monotone	Redundant	Stable	Interpretable
vertex / edge connectivity	✗		✗	✓
avg. shortest distance	✗	✗	✗	✓
diameter	✗	✗	✗	✓
efficiency	✓	✗	✓	✓
[redacted]				

Our choice: natural connectivity as a reliable robustness measure

Measures	S. Monotone	Redundant	Stable	Interpretable
avg. available flows		✓	✗	✓
algebraic connectivity	✗		✗	✗
effective resistance	✓	✓	✓	✓
number of spanning trees	✗		✗	
spectral radius / gap			✓	✗
natural connectivity	✓	✓	✓	✓

Today's Roadmap

- Network Robustness
  - Intro
  - Main Questions
- Measuring Robustness Q1
- Modifying Robustness Q2

Natural connectivity

Stony Brook University Computer Science CUNY

$$\bar{\lambda} = \ln\left(\frac{1}{n}\sum_{j=1}^{n} e^{\lambda_j}\right) = \ln\left(\frac{1}{n}S(G)\right)$$

- "average" eigenvalue of the graph

$$S(G) = \sum_{i=1}^{n} SC(i) = \sum_{i=1}^{n}\sum_{k=0}^{\infty}\frac{(\mathbf{A}^k)_{ii}}{k!} = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{u_{ij}}^2 e^{\lambda_j}$$

$$= \sum_{j=1}^{n} e^{\lambda_j}\sum_{i=1}^{n}\mathbf{u_{ij}}^2 = \sum_{j=1}^{n} e^{\lambda_j}$$

- Interpretation:
  - weighted sum of closed walks in the graph

J. Wu, B. Mauricio, Y.-J. Tan, and H.-Z. Deng. Natural connectivity of complex networks. *Chinese Physics Letters*, 27(7):78902, 2010.

Main questions [Chan, Akoglu, Tong SDM'14]

Stony Brook University Computer Science

CUNY

Q1 How to measure the robustness of a given network?

➡ natural connectivity

Q2 How to modify a given network to improve its robustness?

➡ optimally, rather than ad-hoc

3 Manipulation Problems

Given: A large network G (with $n \times n$ adjacency matrix $\mathbf{A}$) and an integer (budget) k;

PROBLEM 1. **MIOBI-BREAKEDGE (Edge Deletion)**

Output: A set of k *edges* from $\mathbf{A}$, the deletion of which creates the largest *drop* of the network robustness

PROBLEM 2. **MIOBI-BREAKNODE (Node Deletion)**

Output: A set of k *nodes* from $\mathbf{A}$, the deletion of which creates the largest *drop* of the network robustness

PROBLEM 3. **MIOBI-MAKEEDGE (Edge Addition)**

Output: A set of k *non-edges* of $\mathbf{A}$, the addition of which creates the largest *increase* in the network robustness

Problem hardness (1)

- Node deletion is **NP-hard**
- Basic idea: reduction from P1 (known NP-hard)
- P1 (k-independent set): are there k nodes no two of which are adjacent?
- P2 (k node deletion): are there k nodes deletion of which makes all eigenvalues ≤ 0?

$$A = \begin{bmatrix} S_{kxk} & X_{(k)x(n-k)} \\ X_{(k)x(n-k)} & T_{(n-k)x(n-k)} \end{bmatrix}$$

Proof #1: If YES to P1(*G*,*k*) → YES to P2(*G*, *n-k*)

$$\text{YES to P1} \Rightarrow S_{kxk} = \mathbf{0} \xrightarrow[\text{Nodes in } T]{\text{Removing}} \lambda(\tilde{A}) = \lambda(\mathbf{0}) = 0 \Rightarrow \text{YES to P2}$$

Proof #2: If NO to P1(*G*,*k*) → NO to P2(*G*, *n-k*)

$$\text{Suppose YES to P2} \xrightarrow[\text{Nodes in } T]{\text{Removing}} \lambda(\tilde{A}) = \lambda(\mathbf{0}) \le 0 \xrightarrow{S(i,j) \ge 0}$$

$$\Rightarrow S_{kxk} = \mathbf{0} \Leftrightarrow \text{Nodes in } \boldsymbol{S} \text{ being ind. set} \Rightarrow \text{contradict}$$

Problem hardness (2)

- Edge deletion is **NP-hard***
- Basic idea: reduction from P1 (known NP-hard)
- P1 (Hamiltonian Path): is there a path that visits every node exactly once?
- P2 (k edge deletion): are there k edges deletion of which makes the largest eigenvalue ≤ $\alpha = 2\cos(\frac{\pi}{n+1})$?

Proof #1: If YES to P1(G, k) → YES to P2(G, e-k)

YES to P1 ➡ Remove non-HP edges ➡ $\lambda_1(P_n) = 2\cos(\frac{\pi}{n+1})$

Proof #2: If NO to P1(G, k) → NO to P2(G, e-k)

Hardness of edge addition problem remains to be studied

Network modification

- When the nodes/edges are modified, let $\bar{\lambda}_\Delta$ denote updated robustness

$$\bar{\lambda}_\Delta = \ln(\frac{1}{n}\sum_{j=1}^{n} e^{\lambda_j+\Delta\lambda_j})$$

min./max. $e^{\lambda_1+\Delta\lambda_1} + e^{\lambda_2+\Delta\lambda_2} + \ldots + e^{\lambda_n+\Delta\lambda_n}$

$$e^{\lambda_1}(e^{\Delta\lambda_1} + e^{(\lambda_2-\lambda_1)}e^{\Delta\lambda_2} + \ldots + e^{(\lambda_n-\lambda_1)}e^{\Delta\lambda_n})$$

$$c_1(e^{\Delta\lambda_1} + c_2e^{\Delta\lambda_2} + \ldots + c_ne^{\Delta\lambda_n})$$

where $c_1 = e^{\lambda_1}$ and $c_j = e^{(\lambda_j-\lambda_1)}$ for $2 \leq j \leq t$

Network modification

Stony Brook University
Computer Science
CUNY

- When the nodes/edges are modified, let $\bar{\lambda}_\Delta$ denote updated robustness

$$\bar{\lambda}_\Delta = \ln\left(\frac{1}{n}\sum_{j=1}^{n} e^{\lambda_j + \Delta\lambda_j}\right)$$

Updating the eigenvalues.

LEMMA 4.1. *Given a perturbation* $\Delta\mathbf{A}$ *to a matrix* $\mathbf{A}$*, its eigenvalues can be updated by*

(4.3) $$\Delta\lambda_j = \mathbf{u_j}'\Delta\mathbf{A}\mathbf{u_j}.$$

Updating the eigenvectors.

LEMMA 4.2. *Given a perturbation* $\Delta\mathbf{A}$ *to a matrix* $\mathbf{A}$*, its eigenvectors can be updated by*

(4.7) $$\Delta\mathbf{u_j} = \sum_{i=1, i\neq j}^{n}\left(\frac{\mathbf{u_i}'\Delta\mathbf{A}\mathbf{u_j}}{\lambda_j - \lambda_i}\mathbf{u_i}\right).$$

Network modification

PROBLEM 1. **MIoBI-BREAKEDGE (Edge Deletion)**

- Deleting an edge from A: $\Delta\mathbf{A}(p,r) = \Delta\mathbf{A}(r,p) = -1$

$$\min_{(p,r)\in E} \; c_1\left(e^{-2\mathbf{u_{p1}}\mathbf{u_{r1}}} + c_2 e^{-2\mathbf{u_{p2}}\mathbf{u_{r2}}} + \ldots + c_n e^{-2\mathbf{u_{pn}}\mathbf{u_{rn}}}\right)$$

PROBLEM 2. **MIoBI-BREAKNODE (Node Deletion)**

- Deleting a node from A: $(i,v) = (v,i) = -1, v \in \mathcal{N}(i)$

$$\min_{i\in V} \; c_1\left(e^{-2\mathbf{u_{i1}}\sum_{v\in N(i)}\mathbf{u_{v1}}} + \ldots + c_n e^{-2\mathbf{u_{in}}\sum_{v\in \mathcal{N}(i)}\mathbf{u_{vn}}}\right)$$

PROBLEM 3. **MIoBI-MAKEEDGE (Edge Addition)**

- Adding an edge to A:

$$\max_{\substack{(p,r)\notin E \\ p\in V, r\in V}} \; c_1\left(e^{2\mathbf{u_{p1}}\mathbf{u_{r1}}} + c_2 e^{2\mathbf{u_{p2}}\mathbf{u_{r2}}} + \ldots + c_n e^{2\mathbf{u_{pn}}\mathbf{u_{rn}}}\right)$$

Algorithm outline

- Compute top t eigenpairs
- S = $\emptyset$
- For 1 to k
  - select node/edge* that optimizes respective function, add to S
  - update graph and adj. matrix A
  - update eigenpairs
- Return S

* For edge addition, we consider $O(d^2)$ candidates, for top-d nodes with highest u_1 entry (eigen-vector centrality)

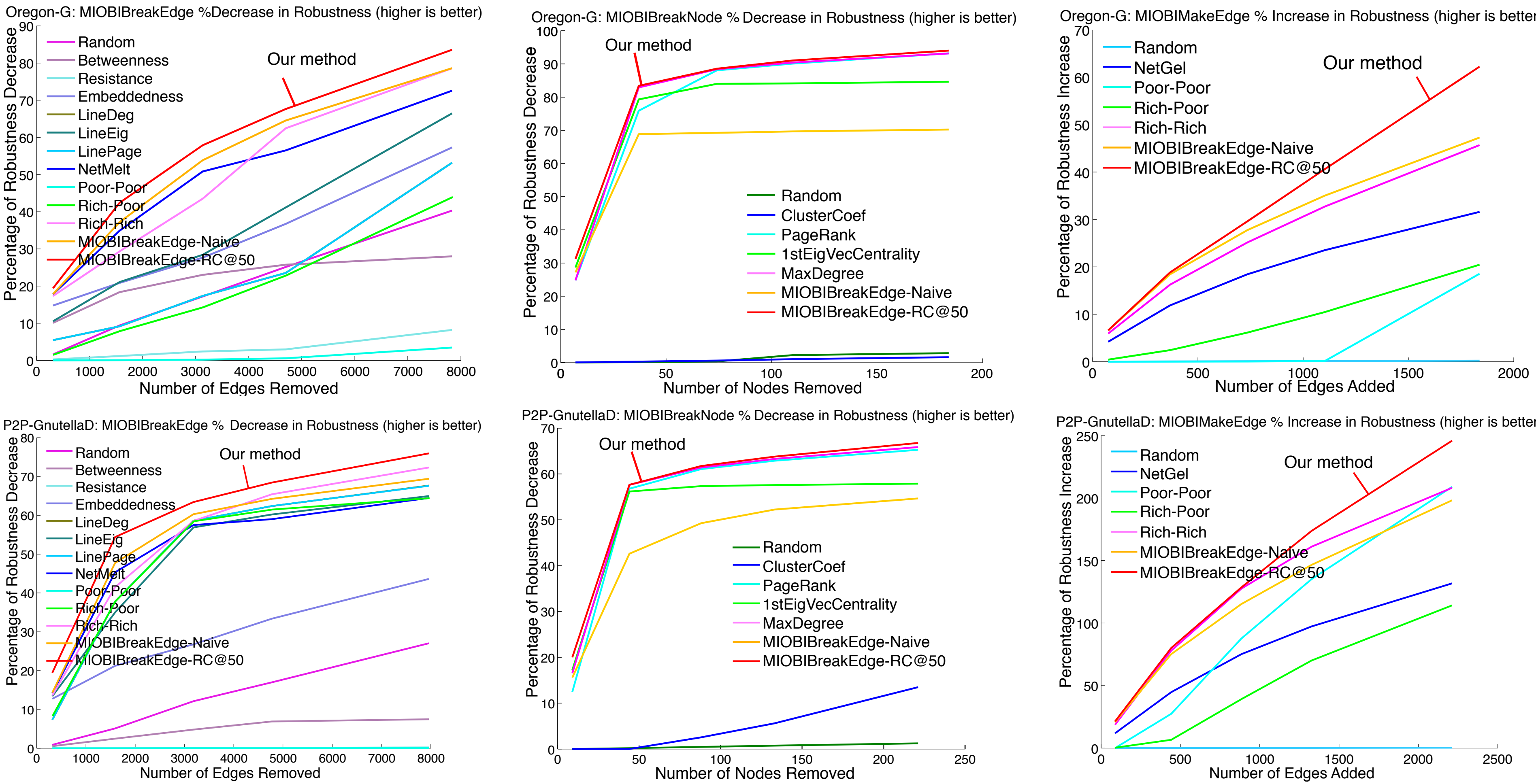
Experiments on real-world graphs

Dataset	n	m
Oregon-A	633	1,086
Oregon-B	1,503	2,810
Oregon-C	2,504	4,723
Oregon-D	2,854	4,932
Oregon-E	3,995	7,710
Oregon-F	5,296	10,097
Oregon-G	7,352	15,665
Oregon-H	10,860	23,409
Oregon-I	13,947	30,584
P2P-GnutellaA	6,301	20,777
P2P-GnutellaB	8,114	26,013
P2P-GnutellaC	8,717	31,525
P2P-GnutellaD	8,846	31,839
P2P-GnutellaE	10,876	39,994

Competing (heuristic) strategies

Edge Deletion	Node Deletion	Edge Addition
random	random	random
rich-rich: max. $d_p d_r$	max. degree	rich-rich: max. $d_p d_r$
poor-poor: min. $d_p d_r$	eig. centrality	poor-poor: min. $d_p d_r$
rich-poor: max. $\|d_p - d_r\|$	pagerank	rich-poor: max. $\|d_p - d_r\|$
betweenness	local clustering	max. $u_{p1} u_{r1}$
embeddedness		
effective resistance		
highest $u_{p1} u_{r1}$		
lineG-degree		
lineG-eig. centrality		
lineG-pagerank		

Robustness change (%) by k



(a) MIOBI-BREAKEDGE (b) MIOBI-BREAKNODE (c) MIOBI-MAKEEDGE

Robustness change (%)

Methods	*O-A*	*O-B*	*O-C*	*O-D*	*O-E*	*O-F*	*O-G*	*O-H*	*O-I*	*G-A*	*G-B*	*G-C*	*G-D*	*G-E*
#Edges removed	543	1405	2362	2466	3855	5049	7833	11705	15292	5194	6503	7881	7960	9999
Random	41.16	42.02	37.48	37.18	39.31	38.45	40.32	40.38	39.65	24.71	24.86	28.00	27.05	27.79
Betweenness	29.49	24.53	24.41	23.56	24.74	25.53	27.99	28.44	29.12	4.51	3.62	13.82	7.46	11.00
Resistance	13.63	11.99	11.40	10.53	10.15	8.47	8.21	7.75	6.71	0.10	0.08	0.35	0.22	0.89
Embeddedness	54.76	56.04	52.92	55.03	56.63	55.82	57.31	60.50	61.07	67.90	62.60	45.24	43.62	26.82
LineDeg	41.10	42.52	43.67	57.95	54.63	55.33	53.18	64.58	67.91	73.94	74.24	63.57	67.67	50.29
LineEig	60.44	61.01	60.45	63.05	63.62	63.20	66.46	67.20	68.54	72.48	72.10	62.31	64.97	49.45
LinePage	41.10	42.52	43.67	57.95	53.77	55.33	53.18	64.58	66.17	73.91	74.15	63.71	67.60	50.00
NetMelt	60.99	61.22	61.38	73.17	68.33	68.86	72.58	75.36	72.47	71.58	71.51	62.36	64.47	48.78
Poor-Poor	13.97	10.67	8.61	2.66	4.80	3.27	3.53	2.55	2.09	0.02	0.02	0.20	0.09	0.59
Rich-Poor	35.78	38.93	40.93	48.52	49.99	46.81	43.98	57.95	64.05	73.26	72.75	62.25	64.45	45.85
Rich-Rich	63.50	64.35	64.30	74.48	74.95	70.12	75.16	79.07	76.01	75.84	76.11	68.95	70.89	55.80
MIOBI-Naive	57.26	64.84	65.00	66.86	70.59	74.88	78.62	79.59	81.66	75.19	74.60	66.88	69.40	50.01
MIOBI-RC@50	**66.11**	**71.10**	**72.78**	**79.66**	**79.10**	**82.05**	**83.57**	**85.97**	**87.04**	**79.73**	**80.34**	**74.59**	**75.96**	**64.68**

Edge Deletion: when k=0.25m edges removed from each graph

Robustness change (%)

Stony Brook University Computer Science CUNY

Methods	*O-A*	*O-B*	*O-C*	*O-D*	*O-E*	*O-F*	*O-G*	*O-H*	*O-I*	*G-A*	*G-B*	*G-C*	*G-D*	*G-E*
#Nodes removed	16	38	63	71	100	132	184	272	349	158	203	218	221	272
Random	4.21	2.58	3.70	1.92	1.71	1.07	2.85	2.50	1.54	0.43	1.74	5.74	1.24	1.35
ClusterCoef	2.28	2.47	2.31	3.05	2.44	2.60	1.62	1.40	0.92	13.66	11.64	12.16	13.49	0.89
PageRank	**93.06**	91.47	93.69	92.20	92.93	92.22	93.19	93.21	**93.63**	75.26	74.58	62.96	65.29	45.91
1stEigVecCentrality	89.89	86.96	88.59	82.54	81.45	85.57	84.63	85.20	81.93	70.53	68.44	54.23	57.89	34.41
MaxDegree	92.25	90.80	93.44	92.36	92.81	92.35	93.18	93.06	93.55	75.56	74.85	63.52	65.86	46.68
MIoBI-Naive	92.38	82.55	89.82	82.20	83.92	83.30	70.22	71.72	69.03	36.60	51.39	59.65	54.67	38.48
MIoBI-RC@50	92.50	**91.51**	**93.92**	**92.47**	**93.49**	**93.15**	**94.04**	**94.24**	92.08	**76.19**	**75.69**	**64.19**	**66.75**	**47.24**

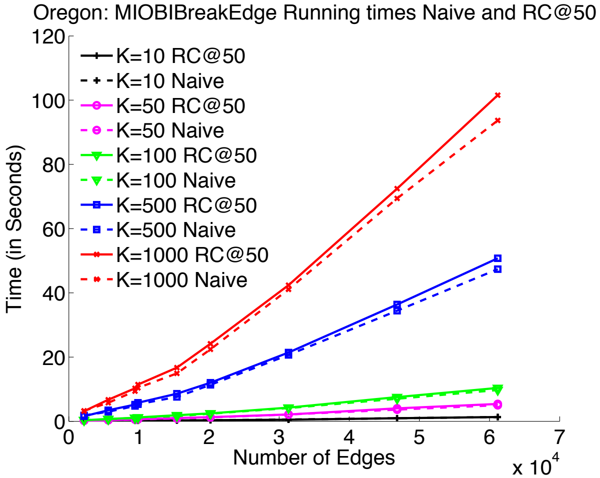
Node Deletion: when k=0.025n nodes removed from each graph

Robustness change (%)

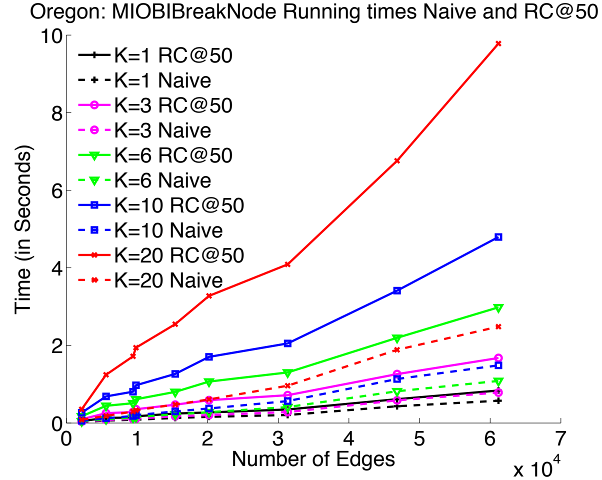
Methods	*O-A*	*O-B*	*O-C*	*O-D*	*O-E*	*O-F*	*O-G*	*O-H*	*O-I*	*G-A*	*G-B*	*G-C*	*G-D*	*G-E*
#Edges added	6	15	25	29	40	53	74	109	139	63	81	87	88	109
Random	0.03	0.01	0.03	0.01	0.15	0.01	0.01	0.01	0.01	0.02	0.00	0.01	0.04	0.05
NetGel	1.79	2.70	2.42	2.40	2.58	3.57	4.20	4.73	4.98	5.48	6.84	12.29	11.84	18.85
Poor-Poor	0.02	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.38
Rich-Poor	0.55	0.47	0.58	0.77	0.64	0.55	0.42	0.44	0.45	0.26	0.22	0.08	0.31	1.58
Rich-Rich	3.38	3.97	3.58	3.86	3.66	5.24	5.94	6.76	6.86	9.58	12.52	14.94	18.70	24.29
MIOBI-Naive	**3.49**	**4.37**	**4.10**	**4.05**	**4.14**	5.60	6.59	7.36	7.75	10.38	13.13	22.62	20.74	34.25
MIOBI-RC@50	**3.49**	**4.37**	**4.10**	**4.05**	**4.14**	**5.62**	**6.61**	**7.41**	**7.81**	**10.49**	**13.20**	**23.16**	**21.40**	**35.84**

Edge Addition: when k=0.01n edges added to each graph

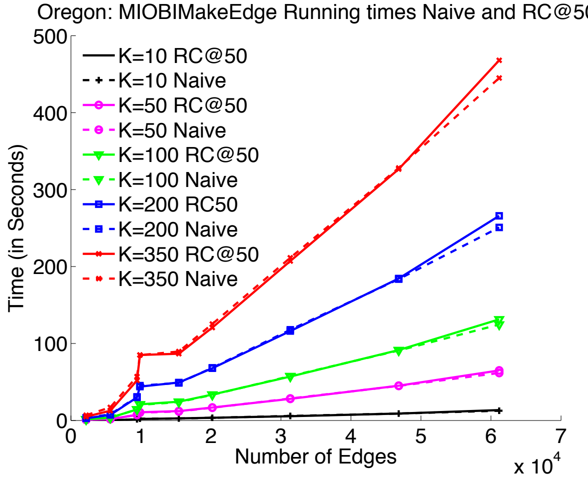
Running time / Scalability



(a) MIOBI-BREAKEDGE



(b) MIOBI-BREAKNODE



(c) MIOBI-MAKEEDGE

Near-linear scalability by network size (#edges)

Today's Roadmap

- Network Robustness
  - Intro
  - Main Questions
- Measuring Robustness Q1
- Modifying Robustness Q2
- Summary

Summary

Q1 How to measure the robustness of a given network?

Q2 How to modify a given network to improve its robustness?

Leman's note: Turn the above into summary statements and conclude.

Thank you!

hauchan@cs.stonybrook.edu

http://www.cs.stonybrook.edu/~datalab/