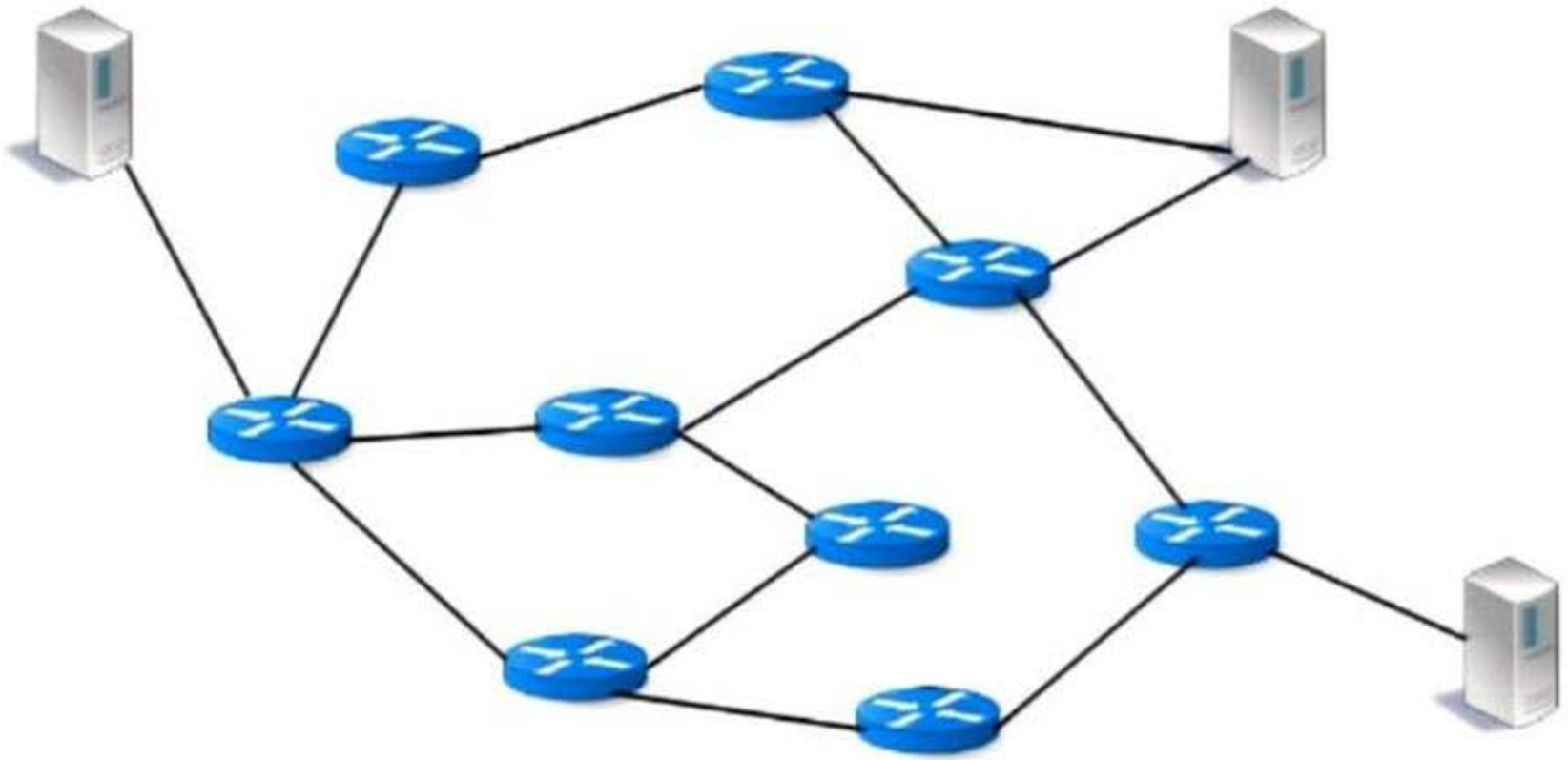
Spanning-edge Centrality: Large-scale computations and applications

C. Mavroforakis, R. Garcia-Lebron,
I. Koutis, **E. Terzi**

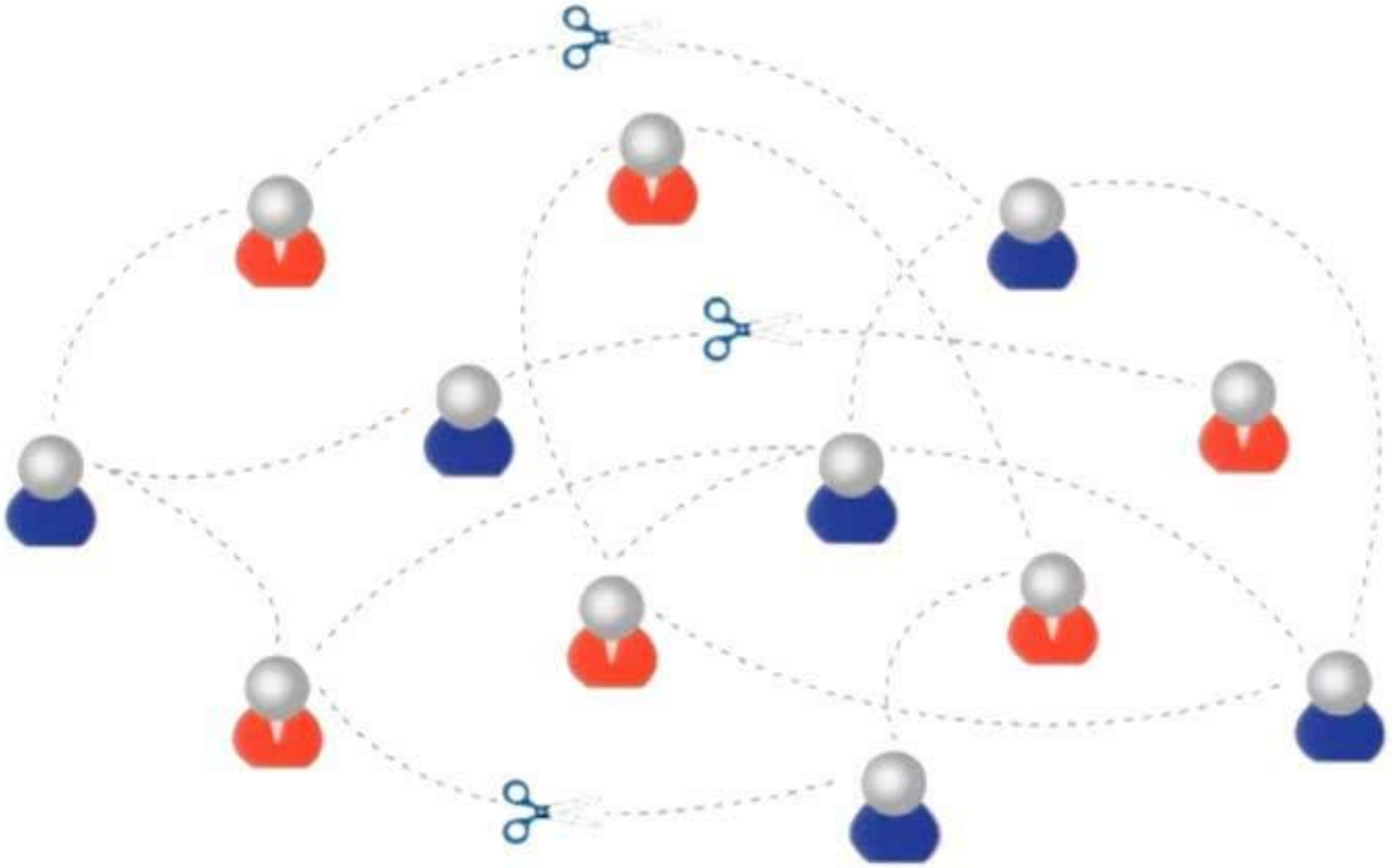
BOSTON UNIVERSITY

Question

- Given a graph $G = (V, E, w)$ what is the importance of every edge $e \in E$?



BOSTON
UNIVERSITY



BOSTON UNIVERSITY

Talk outline

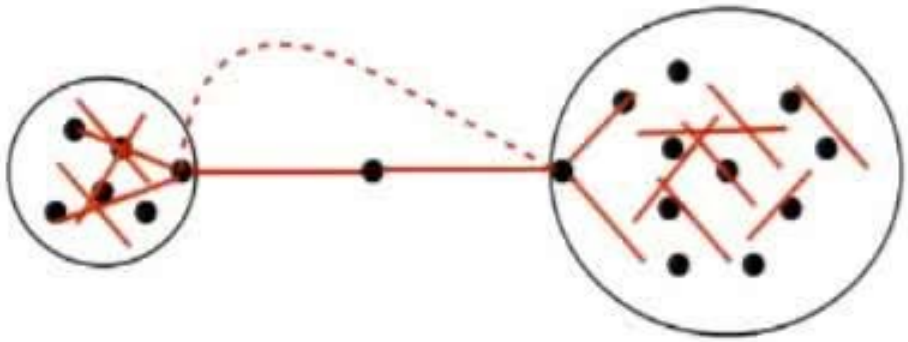
- Examples of edge importance measures
- Spanning-edge centrality and efficient computation
- A framework for computing electrical measures of centrality
- Comparison and usefulness of different measures
- Broader vision and future work

BOSTON UNIVERSITY

Betweenness centrality

- ([Freeman '70]): Betweenness centrality of edge e: The fraction of all-pairs shortest paths that pass through e
- Running time $O(nm + n^2 \log n)$ [Brandes '01]
  - n: number of nodes, m: number of edges
- Speedups via sampling
  - [Bader *et al.* '07, Brandes&Pich '07, Geisberger *et al.* '08, Riondato & Kornaropoulos '14]

Betweenness centrality and small perturbations



BOSTON UNIVERSITY

Spanning-edge centrality

- ([Teixeira *et al.* '13]) Edge importance in phylogenetic studies
- Given G = (V,E), the StC of edge $e \in E$ is the fraction of spanning trees in which e participates in
- Computation by matrix-tree theorem: $O(mn^{3/2})$
- Connections with *information propagation*

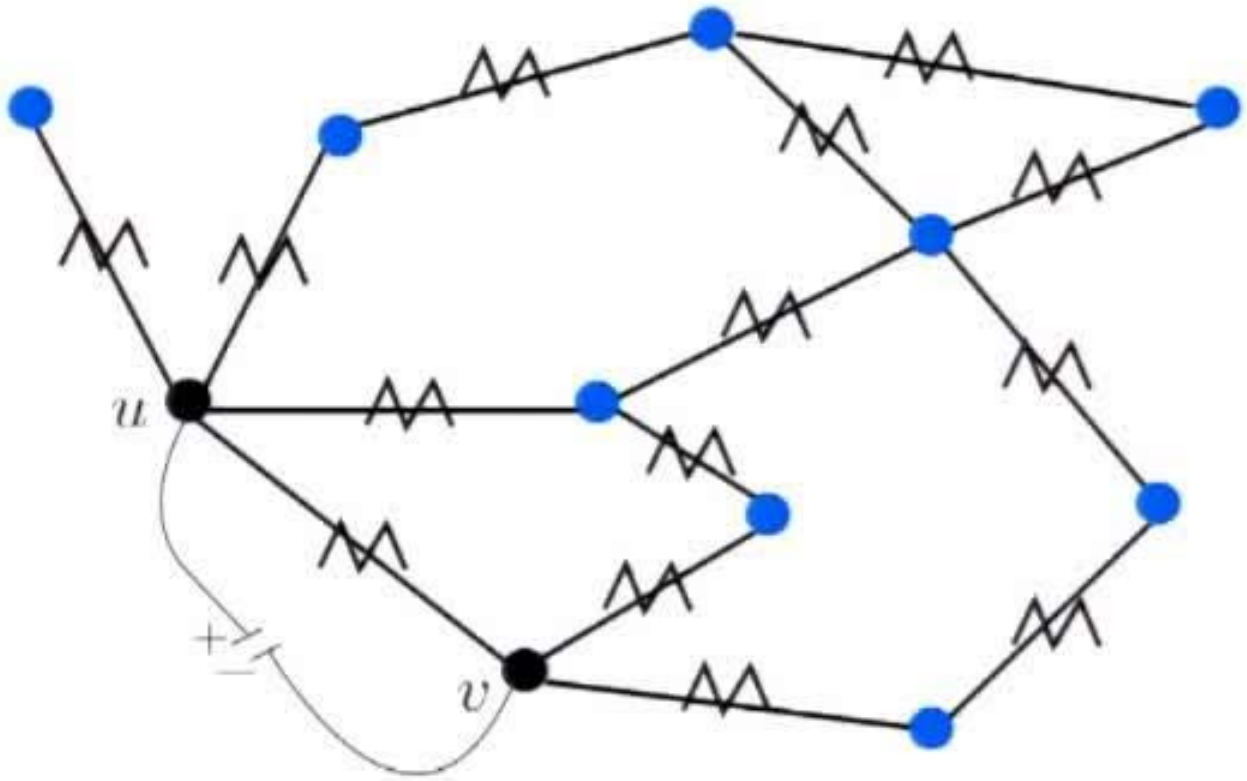
BOSTON UNIVERSITY

Effective resistance

- For every edge $e \in E$

$$R(e) = \mathrm{StC}(e)$$

Effective resistances



BOSTON UNIVERSITY

Computing Effective Resistances (TreeC) [Spielman and Srivastava '08]

Input: $G = (V, E)$.
Output: $R(e)$ for every $e = \{u, v\} \in E$

1: $Z = [\,]$, $L =$ Laplacian of G
2: Construct random projection matrix Q of size $k \times m$
3: Compute $Y = QB$
4: **for** $i = 1 \ldots k$ **do**
5: Approximate z_i by solving: $Lz_i = Y(:, i)$
6: $Z = [Z; z_i^T]$
7: return $R(e) = \|Z(:, u) - Z(:, v)\|_2^2$

Computing Effective Resistances

(TreeC) [Spielman and Srivastava '08]

Input: $G = (V, E)$.
Output: $R(e)$ for every $e = \{u, v\} \in E$

1: $Z = [\,]$. $L =$ Laplacian of G
2: Construct random projection matrix Q of size $k \times m$
3: Compute $Y = QB$
4: **for** $i = 1 \ldots k$ **do**
5: Approximate z_i by solving: $Lz_i = Y(:, i)$ ← [Koutis *et al.* '11]
6: $Z = [Z; z_i^T]$
7: return $R(e) = \|Z(:, u) - Z(:, v)\|_2^2$

BOSTON UNIVERSITY

Computing Effective Resistances

(TreeC) [Spielman and Srivastava '08]

Input: $G = (V, E)$.
Output: $R(e)$ for every $e = \{u, v\} \in E$

1: $Z = [\,]$, $L =$ Laplacian of G
2: Construct random projection matrix Q of size $k \times m$
3: Compute $Y = QB$
4: **for** $i = 1 \ldots k$ **do**
5: Approximate z_i by solving: $Lz_i = Y(:, i)$ ← [Koutis *et al.* '11]
6: $Z = [Z; z_i^T]$
7: return $R(e) = \|Z(:, u) - Z(:, v)\|_2^2$

- Running time: $O\left(m \log^2 n \log \frac{1}{\epsilon}\right)$
- Accuracy: $(1 \pm \epsilon)^2$

BOSTON UNIVERSITY

Speedups (`Fast-TreeC`)

Input: $G = (V, E)$.
Output: $R(e)$ for every $e = \{u, v\} \in E$

1: $L =$ Laplacian of G
2: **for** $i = 1 \ldots k$ **do**
3: Construct a vector q of size $1 \times m$
4: Compute $y = qB$
5: Approximate z by solving: $Lz = y$
6: return $R(e) = R(e) + \|z(u) - z(v)\|_2^2$

Speedups (`Fast-TreeC`)

Input: $G = (V, E)$.
Output: $R(e)$ for every $e = \{u, v\} \in E$

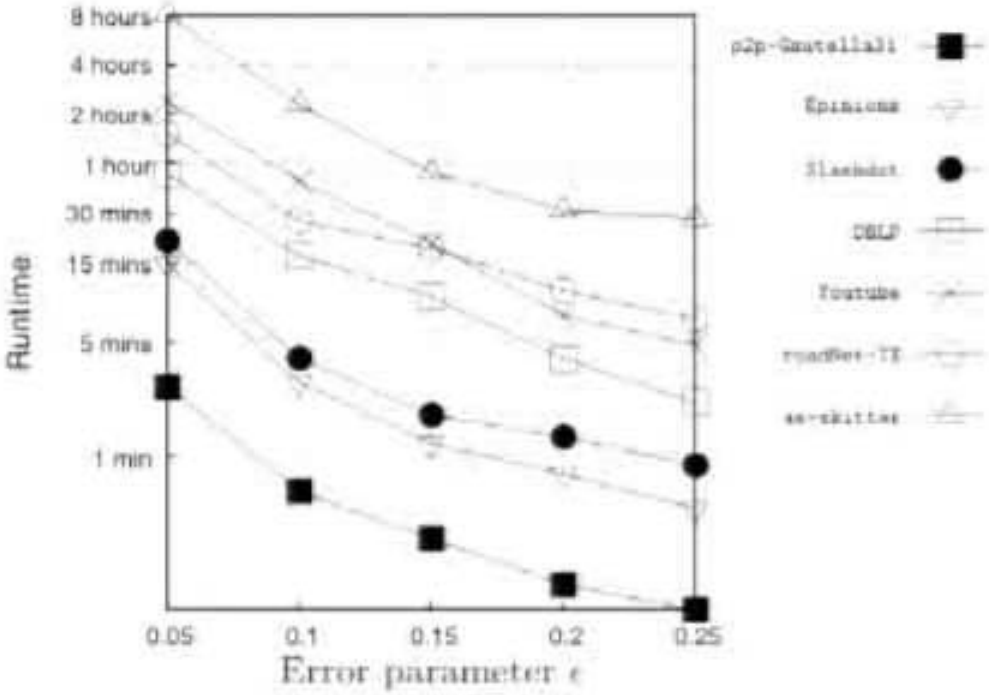
1: $L =$ Laplacian of G
2: **for** $i = 1 \ldots k$ **do** ← parallel
3: Construct a vector q of size $1 \times m$
4: Compute $y = qB$
5: Approximate z by solving: $Lz = y$
6: return $R(e) = R(e) + \|z(u) - z(v)\|_2^2$

Space efficiency

BOSTON UNIVERSITY

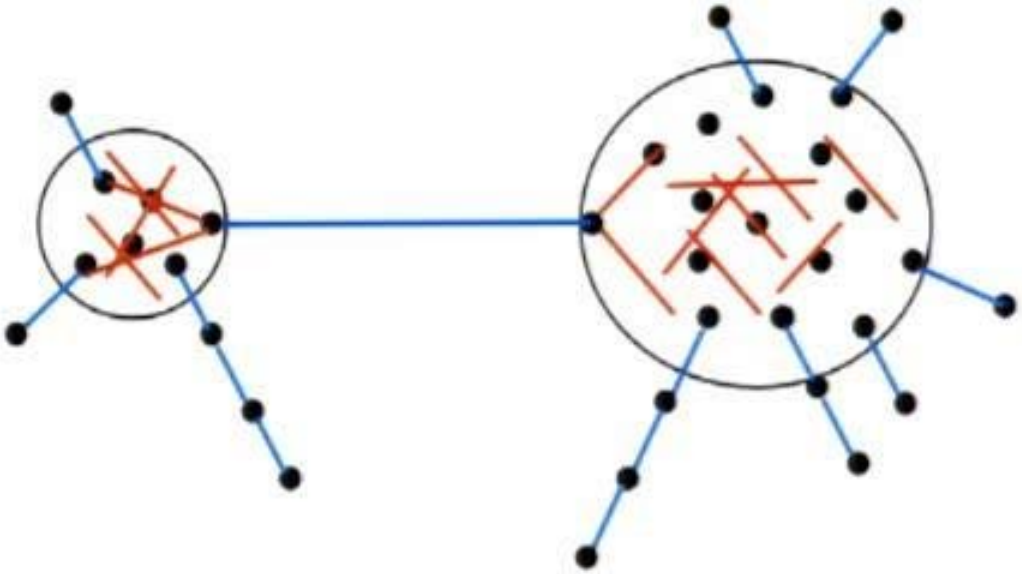
Spanning-tree centrality: experiments

Dataset name	#Nodes in LCC	#Edges in LCC
CrQc	4 158	13 422
p2p-Gnutella08	6 299	20 776
Oregon-1	11 174	23 409
HepTh	8 638	24 806
wiki-Vote	7 066	100 736
p2p-Gnutella31	62 561	147 878
Epinions	75 877	405 739
Slashdot	82 168	504 230
Amazon	334 863	925 872
DBLP	317 080	1 049 866
roadNet-TX	1 351 137	1 878 201
Youtube	1 134 890	2 987 624
as-skitter	1 694 616	11 094 209
Patents	3 764 117	16 511 740



BOSTON UNIVERSITY

Bridges



2-core speedup

- If G=(V,E) is a connected graph and $C_2(G) = (V',E')$ is its 2-core, then for every edge $e \in E$

$$\textsc{StC}\,(e, C_2(G)) = \textsc{StC}(e, G)$$

- Also for every edge $e \in E \setminus E'$

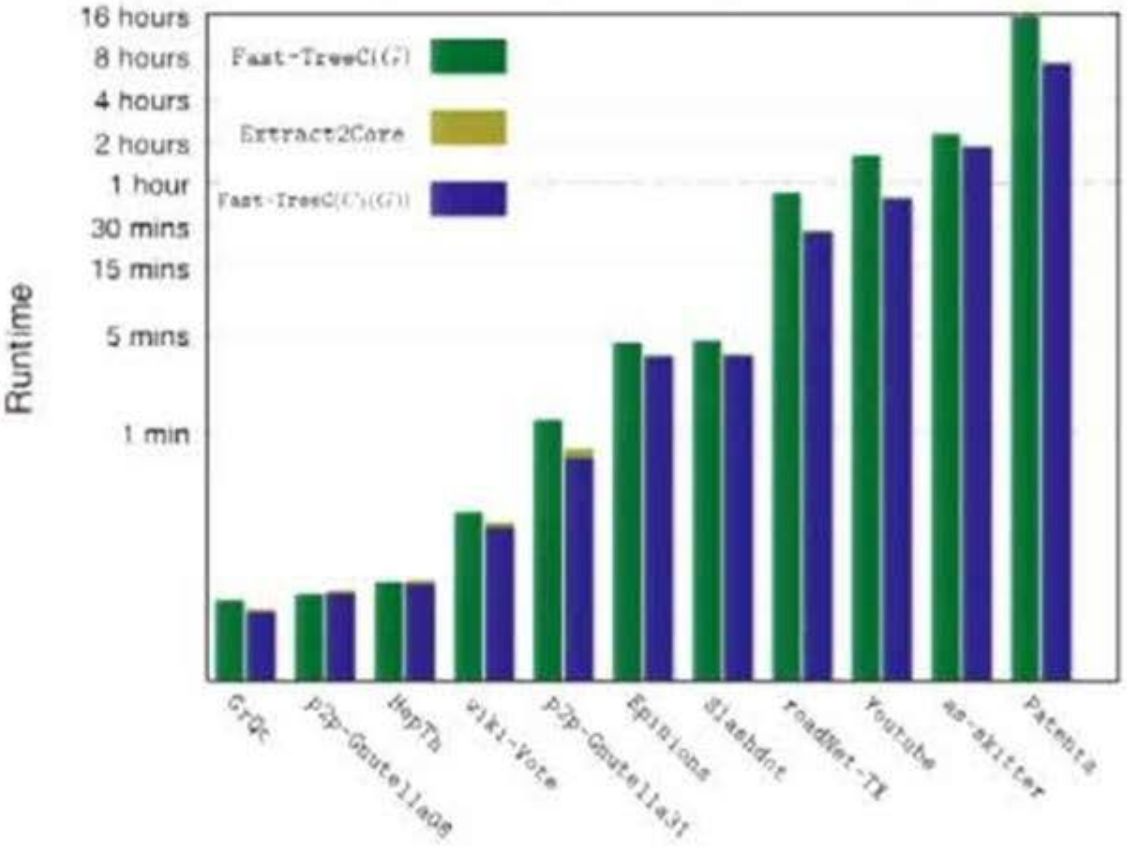
$$\textsc{StC}(e) = 1$$

2-cores of real data

Dataset name	#Nodes in LCC	#Edges in LCC	#Nodes in the 2-Core	#Edges in the 2-Core
CrQc	4 158	13 422	3 413	12 677
p2p-Gnutella08	6 299	20 776	4 535	19 012
Oregon-1	11 174	23 409	7 228	19 463
HepTh	8 638	24 806	7 059	23 227
wiki-Vote	7 066	100 736	4 786	98 456
p2p-Gnutella31	62 561	147 878	33 816	119 133
Epinions	75 877	405 739	37 300	367 162
Slashdot	82 168	504 230	52 181	474 243
Amazon	334 863	925 872	305 892	896 901
DBLP	317 080	1 049 866	271 646	1 004 432
roadNet-TX	1 351 137	1 878 201	1 068 728	1 596 792
Youtube	1 134 890	2 987 624	470 164	2 322 898
as-skitter	1 694 616	11 094 209	1 463 934	10 863 527
Patents	3 764 117	16 511 740	4 093 271	15 840 895

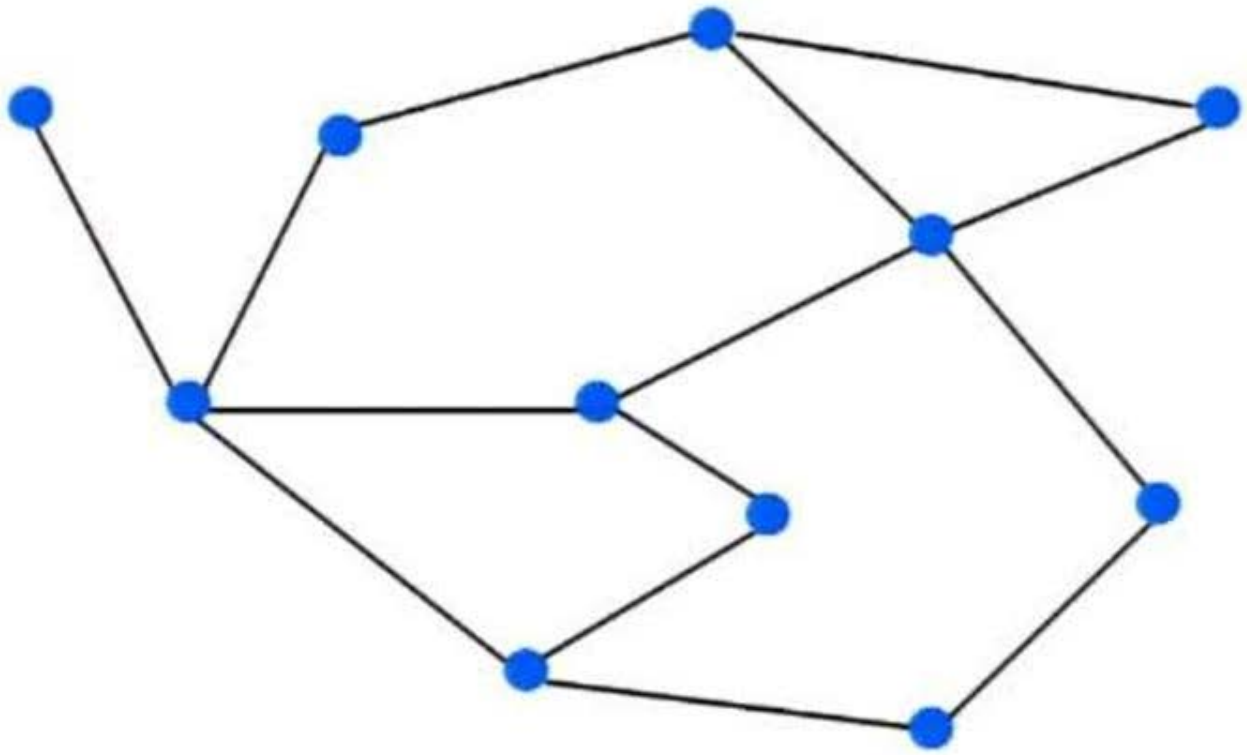
BOSTON UNIVERSITY

2-core speedup in practice



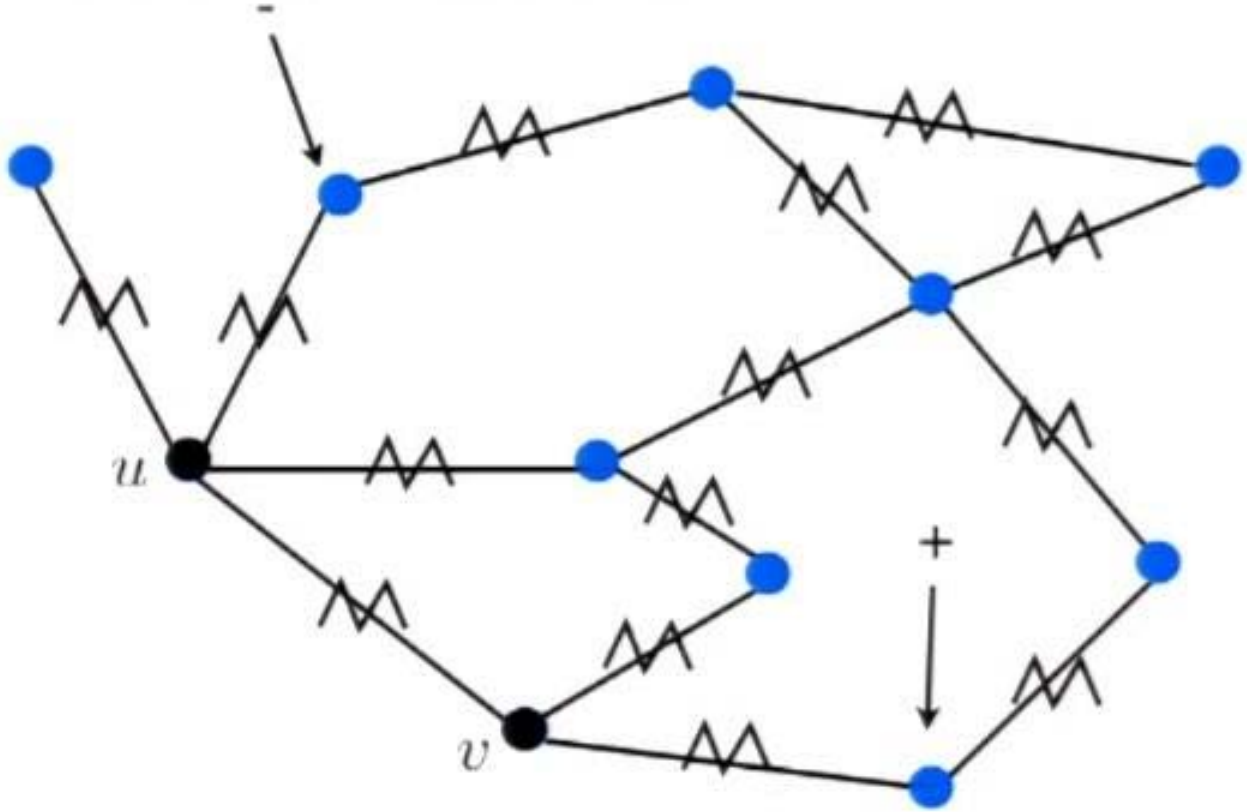
BOSTON UNIVERSITY

Electrical interpretation of StC



BOSTON UNIVERSITY

Electrical measures of edge importance: a general framework



BOSTON UNIVERSITY

Current flow centrality (CfC)

- ([Brandes & Fleischer '05]): Given G=(V,E) the current flow centrality of edge e is the average (s,t)-flow of e, when considering all distinct pairs (s,t)

$$\mathrm{CFC}\left(e=\{u,v\}\right)=\frac{1}{\binom{n}{2}}\sum_{s,t}f_{st}(u,v)$$

- where $f_{st}(u,v)$ is the flow that passes through edge e={u,v} when voltage 1 is applied to endpoints s and t

b-Current flow centrality (b-CfC)

- Given G=(V,E) the current flow centrality of edge e is the average (s,t)-flow of e, when considering all distinct b-tuples of nodes $(s_1, s_2, \ldots, s_b)$

$$b - \mathrm{CFC}\,(e = \{u, v\}) = \frac{1}{|S_b|} \sum_{s_1,\ldots,s_b} f_{s_1,\ldots,s_b}(u, v)$$

- where $f_{st}(u, v)$ is the flow that passes through edge e={u,v} when current 1A exits from every pole $(s_1, s_2, \ldots, s_b)$

Current-flow centrality computation

- For fixed (s,t) pair solve

$$Lx = b$$ [Koutis *et al.* '11]

- with $b(s) = 1,\ b(t) = -1$ otherwise $b(z) = 0$
- set $f_{st}(u, v) = |x(u) - x(v)|$
- approximate $\overline{\mathrm{CFC}}\,(e = \{u, v\}) = \frac{1}{k} \sum_{(s,t) \in S_k} f_{st}(u, v)$

BOSTON UNIVERSITY

Current-flow centrality computation

- For fixed (s,t) pair solve

$$Lx = b$$ [Koutis *et al.* '11]

- with $b(s) = 1,\ b(t) = -1$ otherwise $b(z) = 0$
- set $f_{st}(u, v) = |x(u) - x(v)|$
- approximate $\overline{\mathrm{CFC}}\,(e = \{u, v\}) = \boxed{\frac{1}{k}} \sum_{(s,t) \in S_k} f_{st}(u, v)$

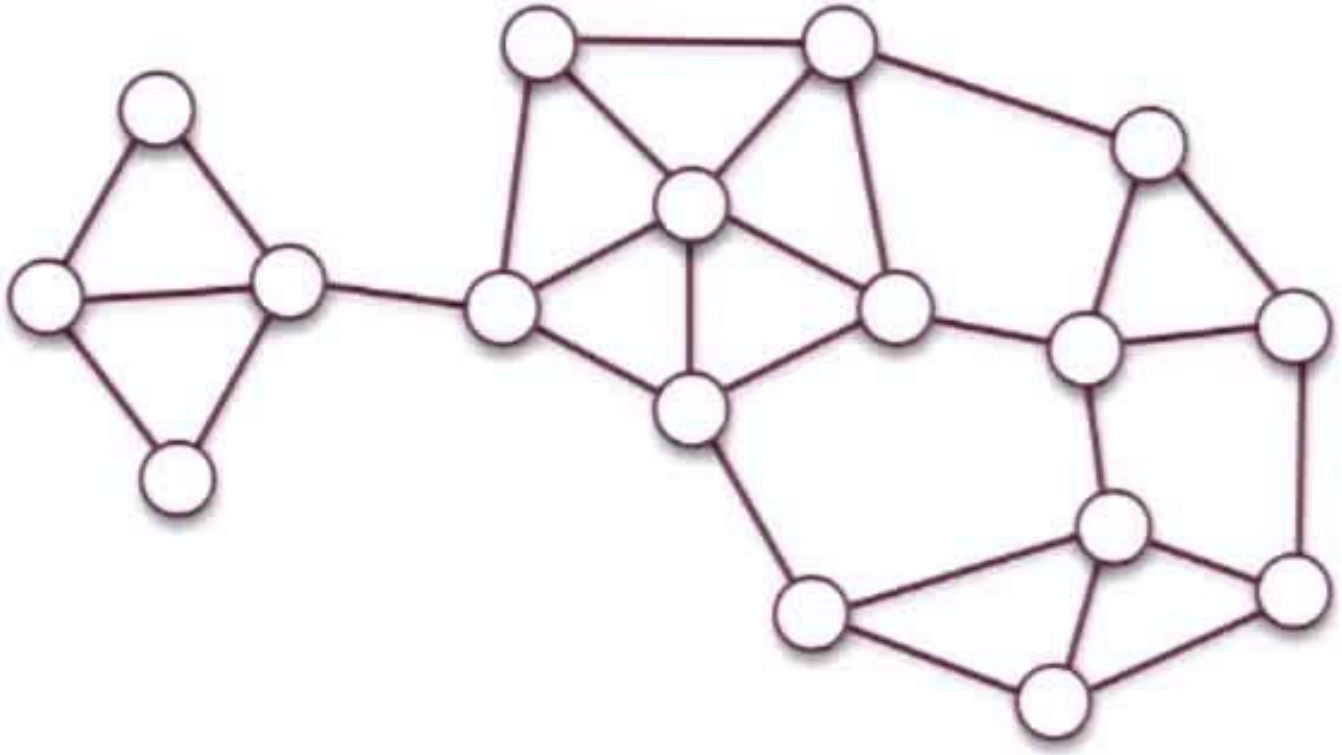
How many samples are enough?

BOSTON UNIVERSITY

Running time till convergence

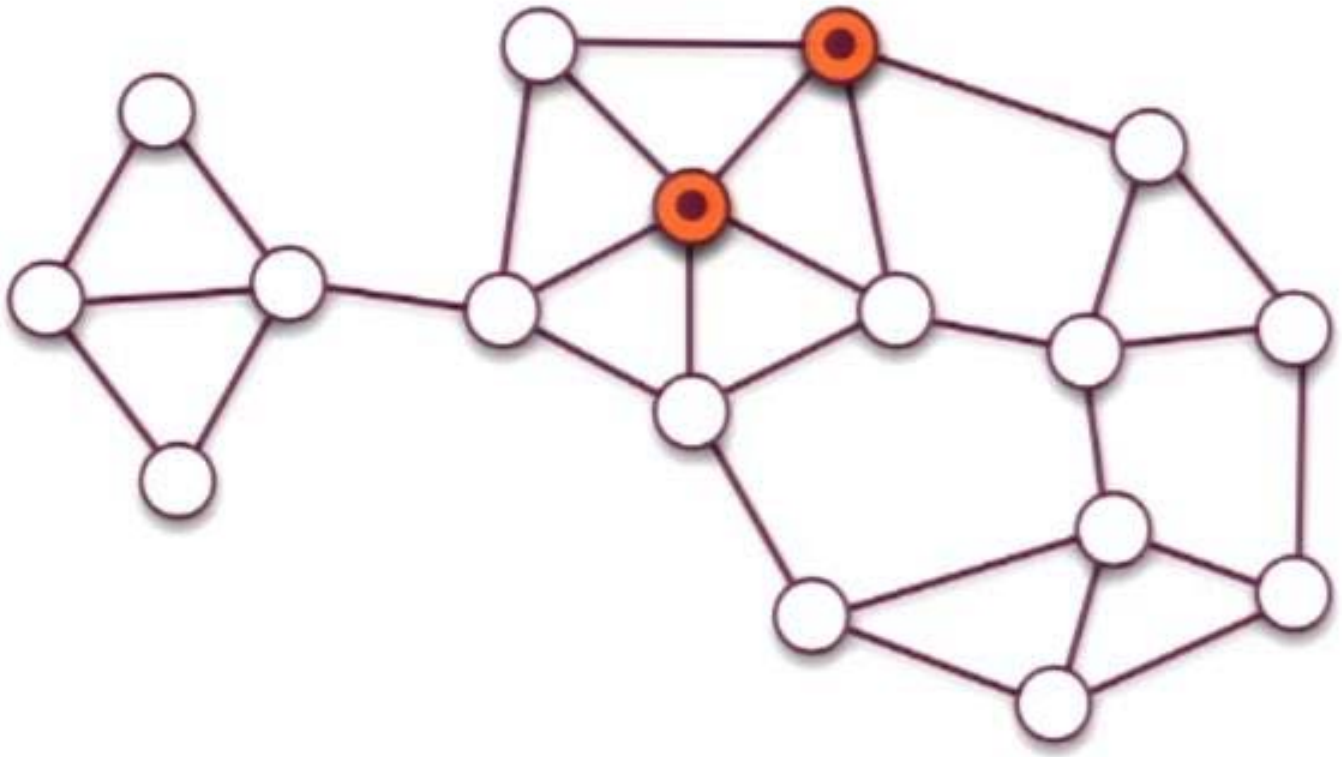
Dataset	Fast-FlowC algorithm ($b = 1$)	($b = 5$)	($b = 20$)	Exact algorithm
GrQc	2.1 mins	1.3 mins	1.3 mins	20 mins
Oregon-1	3.3 mins	1.9 mins	1.4 mins	4h 40mins
Epinions	12h 23mins	5h 26mins	3h	n/a

Information Propagation



independent-cascade model

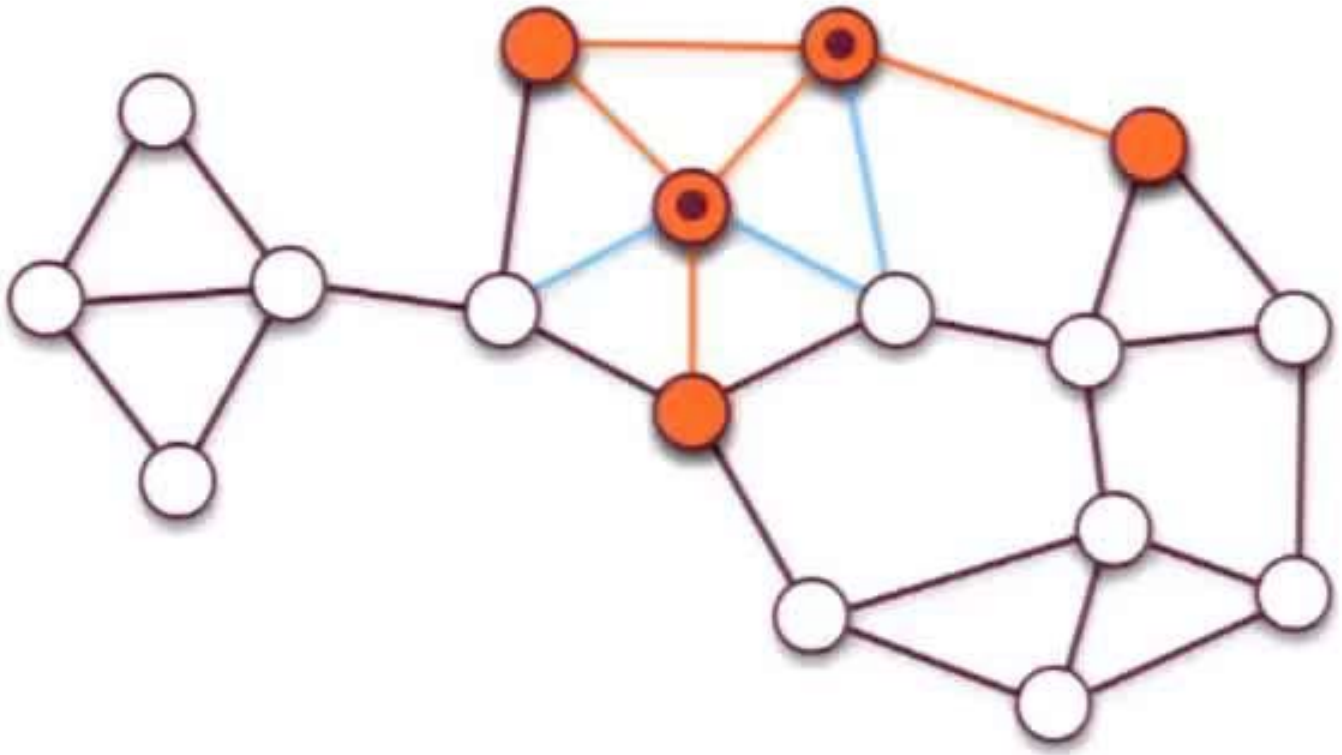
Information Propagation



independent-cascade model

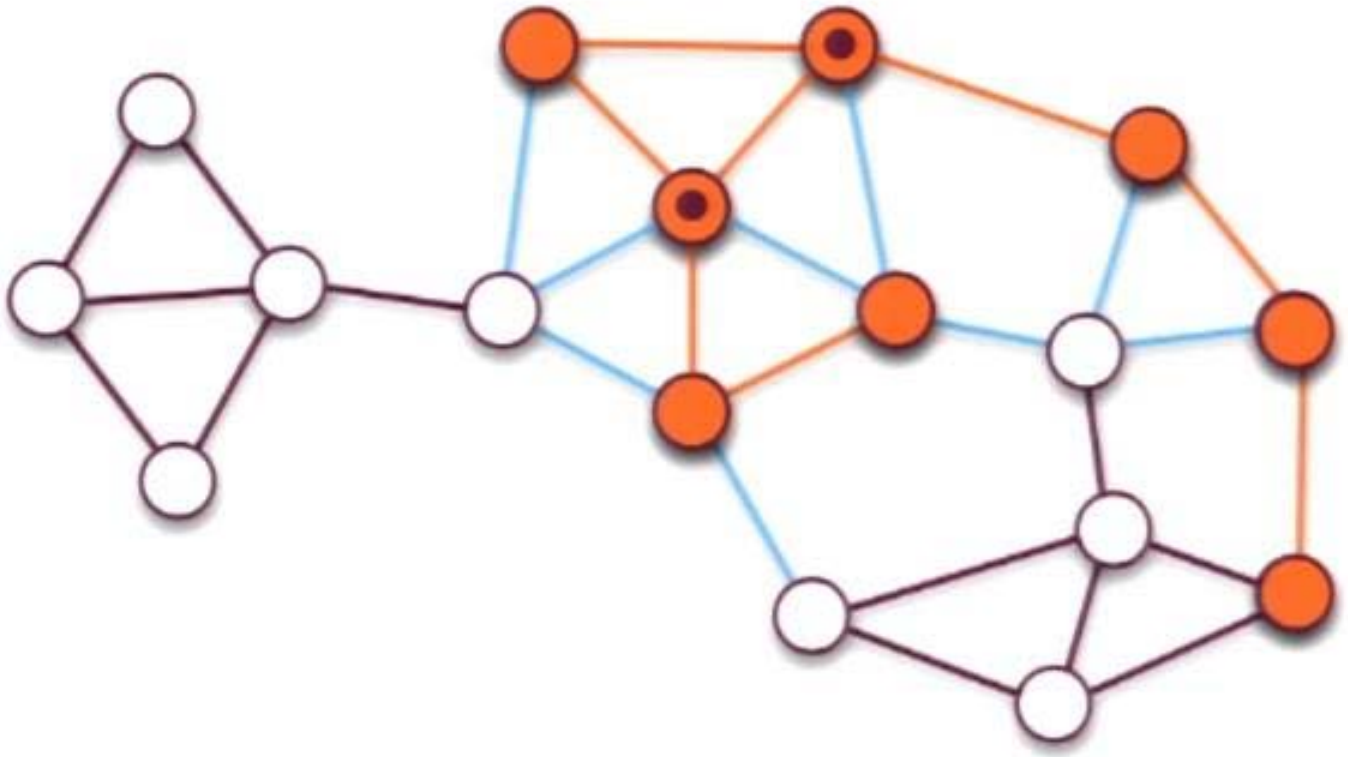
BOSTON UNIVERSITY

Information Propagation



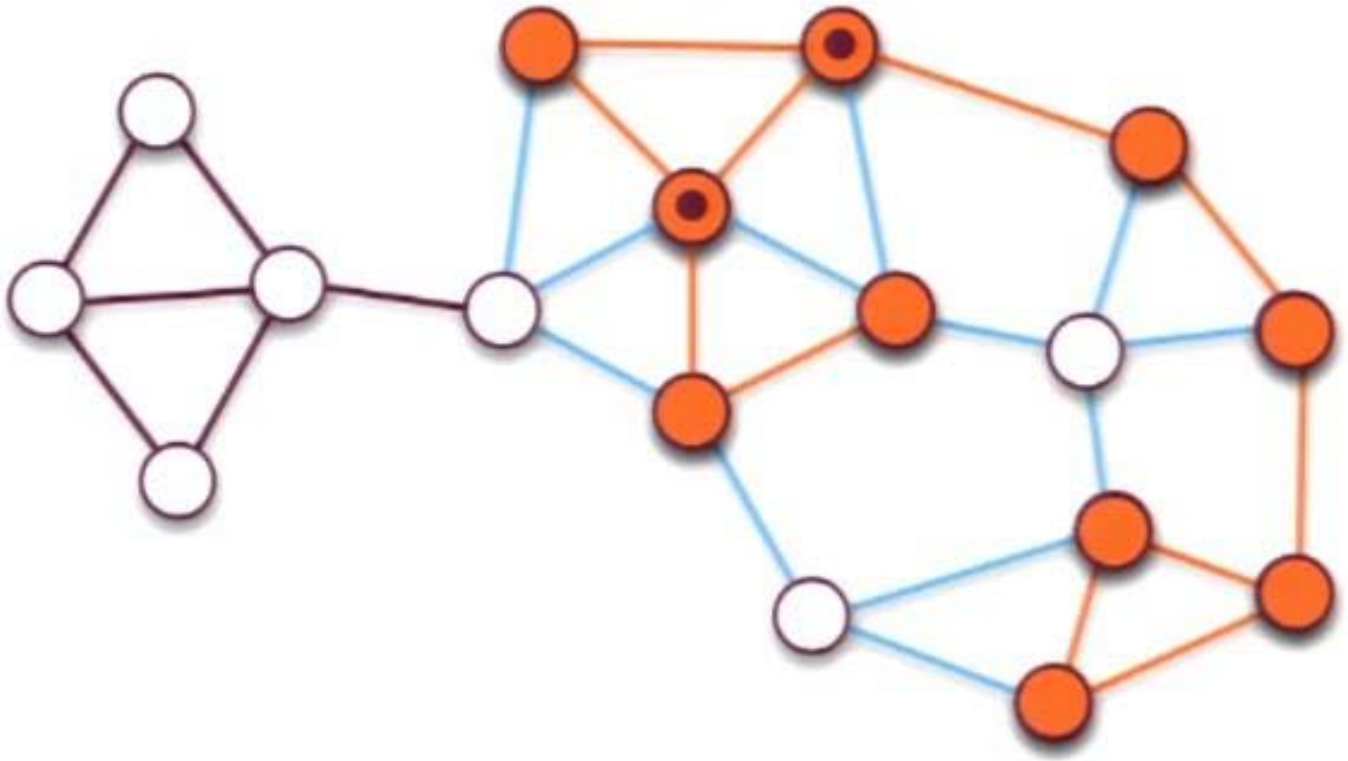
independent-cascade model

Information Propagation



independent-cascade model

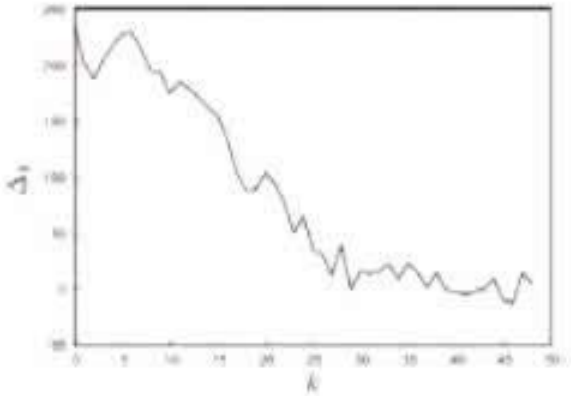
Information Propagation



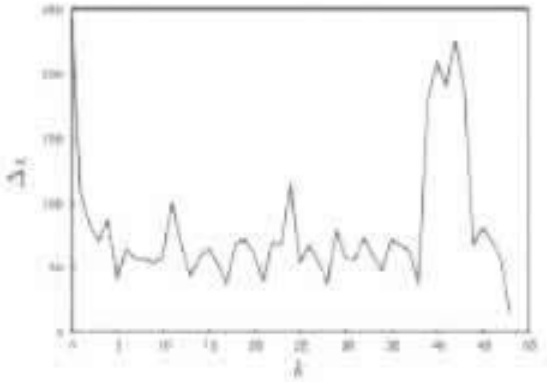
independent-cascade model

BOSTON UNIVERSITY

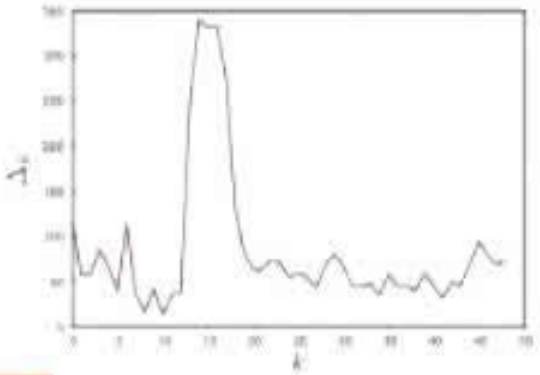
Edge importance and information propagation



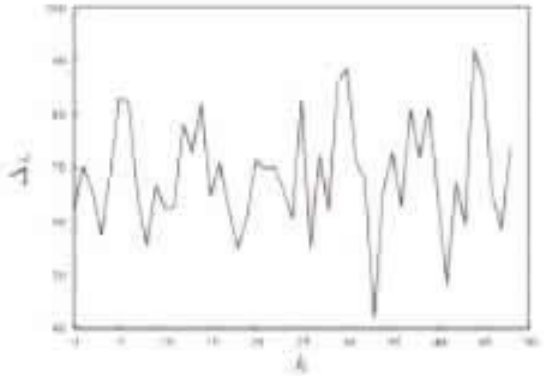
(a) SPANNINGTREE



(b) CURRENTFLOW



(c) BETWEENNESS



(d) RANDOM

BOSTON UNIVERSITY

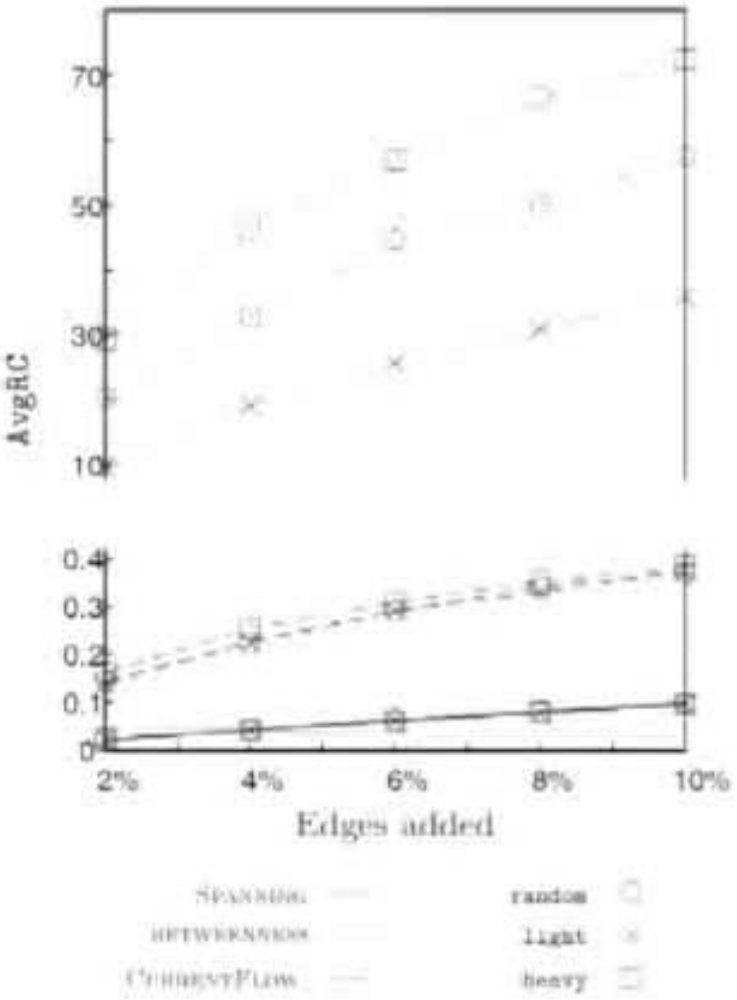
Resilience to noise

- **y**-axis: average relative change in the edge-importance scores.

$$\texttt{AvgRC}(G, G') = \frac{1}{|E \cap E'|} \sum_{e \in E \cap E'} \texttt{RelChange}(e),$$

$$\texttt{RelChange}(e, G, G') = \frac{|c_e - c'_e|}{c_e}$$

- **x**-axis: percentage of noisy edges added
- dataset: Hepth graph (collaboration)



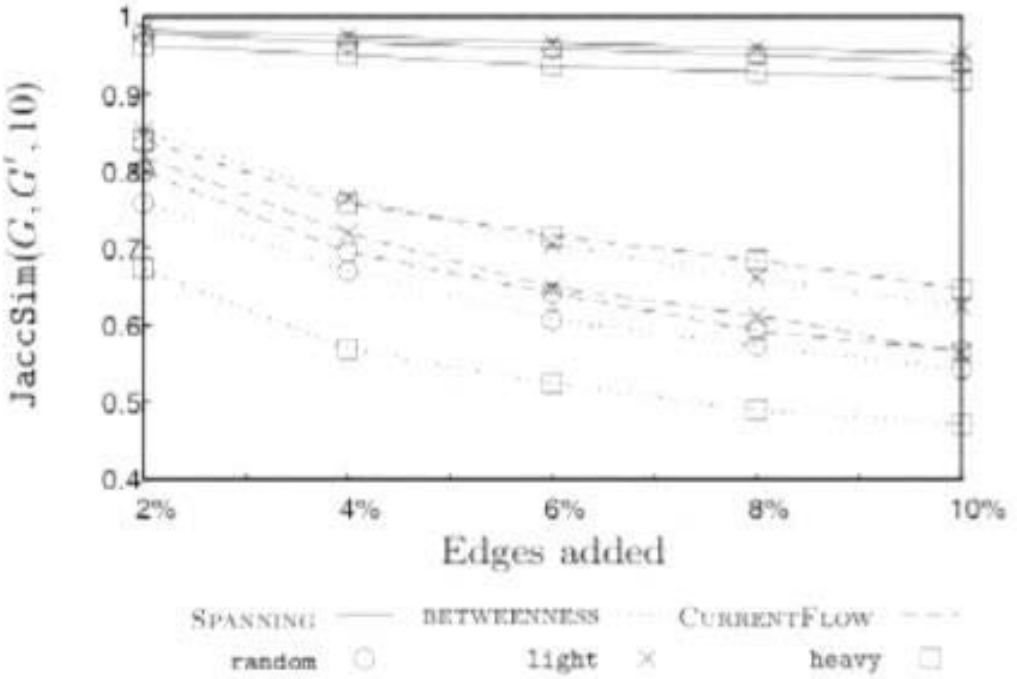
BOSTON UNIVERSITY

Resilience to noise

- **y**-axis: Jaccard similarity between the top-10% scoring edges in the original and noisy graph

$$\texttt{JaccSim}(G, G', k) = \frac{|I(G,k) \cap I(G',k)|}{|I(G,k) \cup I(G',k)|}$$

- **x**-axis: percentage of noisy edges added
- dataset: Hepth graph (collaboration)



BOSTON UNIVERSITY

Long-term goal and vision

- Design edge importance measures that are **intuitive**, **noise-resilient** and **efficiently computable**
  - Optimization + Regularization

- Edge-importance measures for data-mining tasks

BOSTON UNIVERSITY

Thanks!

BOSTON UNIVERSITY