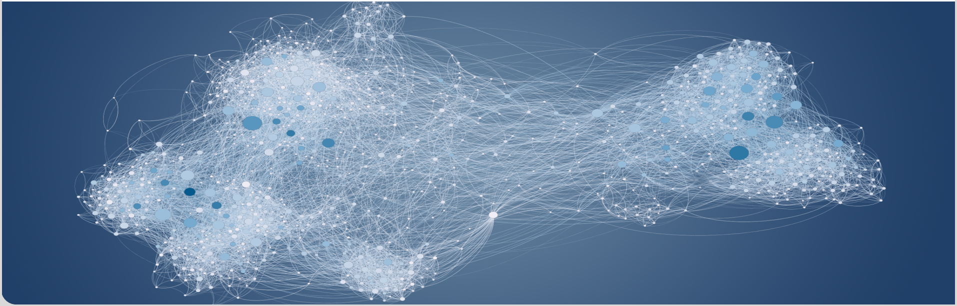
# Estimating Current-Flow Closeness Centrality with a Multigrid Laplacian Solver

E. Bergamini, M. Wegner, D. Lukarski, H. Meyerhenke | October 12, 2016

SIAM WORKSHOP ON COMBINATORIAL SCIENTIFIC COMPUTING (CSC16) – ALBUQUERQUE, NM, USA



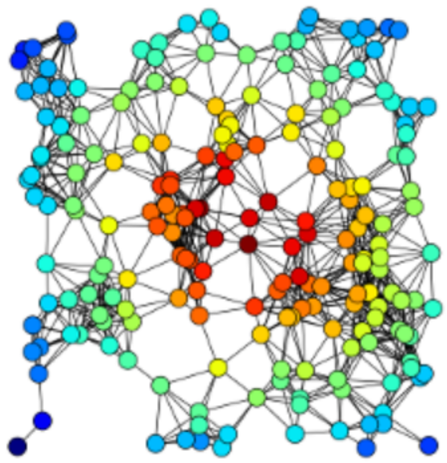
## Overview | Centrality in complex networks

### Network analysis:

- Study structural properties of networks
- Applications: social network analysis, internet, bioinformatics, marketing...

### Centrality

- Ranking nodes
- Closeness centrality: average distance between a node and the others
- Simple and very popular, but
  - assumes information flows through shortest paths only
  - assumes information is inseparable



# Centrality in complex networks

### Electrical closeness

- Information flows through the network like electrical current
- All paths taken into account

However, requires to either invert the Laplacian matrix or solve $n^2$ linear systems

➡ expensive for large networks

### Our contribution

- Two approximation algorithms
- Both require solution of Laplacian linear systems

➡ LAMG implementation in NetworKit

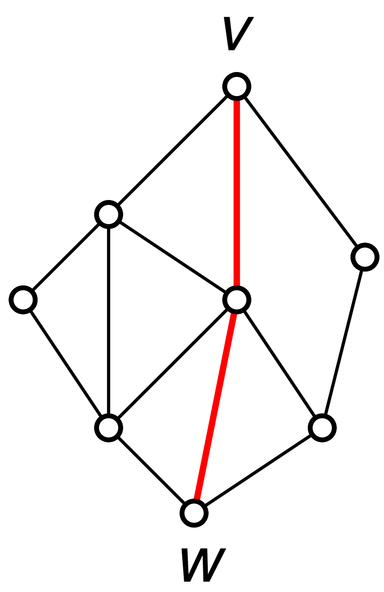
- Properties of electrical closeness and shortest-paths closeness in real-world networks

# Current-flow closeness centrality

### Shortest-path closeness

- Ranks nodes according to average shortest-path distance to other nodes

$$c_{SP}(v) = \frac{n-1}{\sum_{w \in V \setminus \{v\}} d_{SP}(v,w)}$$




- Assumptions on the data

### Current-flow closeness [Brandes and Fleischer, 2005]

- $d_{SP}(v,w)$ replaced with commute time:

$$d_{CF}(v,w) = H(v,w) + H(w,v)$$

- Proportional to potential difference (effective resistance) in electrical network
- All paths are taken into account

# Current-flow closeness centrality

## Current-flow closeness

$$c_{CF}(v) = \frac{n-1}{\sum_{w \in V \setminus \{v\}} d_{CF}(v,w)}$$

## Graph Laplacian

- $L := D - A$
- It can be shown:

$$d_{CF}(v,w) = p_{vw}(v) - p_{vw}(w)$$

where

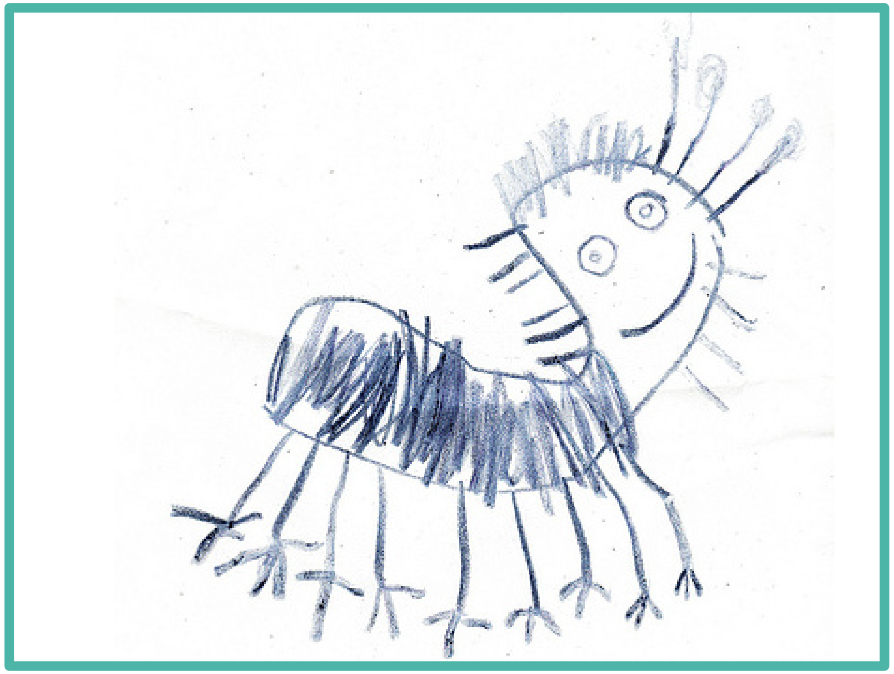
$$Lp_{vw} = b_{vw}$$

$$b_{vw} = \begin{array}{r} \\ \\ v \rightarrow \\ \\ \\ \\ w \rightarrow \\ \\ \\ \end{array} \begin{bmatrix} 0 \\ \dots \\ +1 \\ 0 \\ \dots \\ 0 \\ -1 \\ \dots \\ 0 \end{bmatrix}$$

➡ Solve the system $Lp_{vw} = b_{vw} \;\; \forall w \in V \setminus \{v\}$

- $\Theta(nm \log(1/\tau))$ empirical running time

# Approximation

# Sampling-based approximation

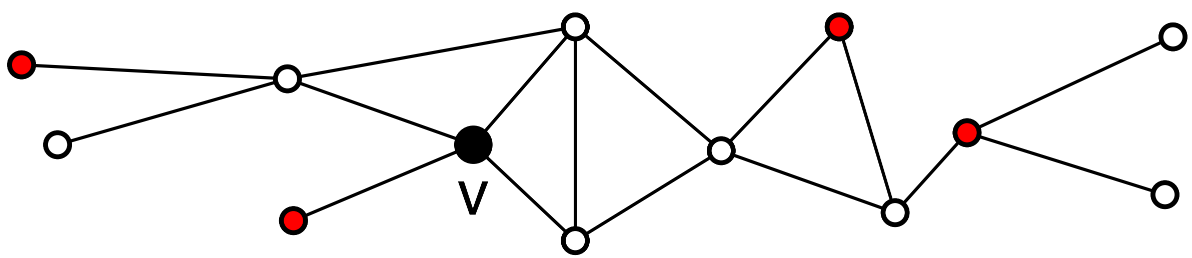
## Current-flow closeness

$$c_{CF}(v) = \frac{n-1}{\sum_{w \in V \setminus \{v\}} p_{vw}(v) - p_{vw}(w)}$$

## Sampling-based approximation

- Set $S = \{s_1, s_2, ..., s_k\}$, $S \subseteq V$
- Approximation:

$$\tilde{c}_{CF}(v) := \frac{k}{n} \cdot \frac{n-1}{\sum_{i=1}^{k} p_{vs_i}(v) - p_{vs_i}(s_i)}$$



# Projection-based approximation

- Johnson- Lindenstrauss Transform:
  - project the system into lower-dymensional space spanned by $\log n/\epsilon^2$ random vectors
  - approximated distances are within $(1+\epsilon)$ factor from exact ones
- Effective resistance $d_{CF}(u, v)$ can be expressed as distances between vectors in $\{W^{1/2}BL^{\dagger}e_u\}_{u \in V}$ **[Spielman, Srivastava, 2011]**

# Projection-based approximation

- Johnson- Lindenstrauss Transform:
  - project the system into lower-dymensional space spanned by $\log n/\epsilon^2$ random vectors
  - approximated distances are within $(1+\epsilon)$ factor from exact ones
- Effective resistance $d_{CF}(u, v)$ can be expressed as distances between vectors in $\{W^{1/2}BL^{\dagger}e_u\}_{u\in V}$ **[Spielman, Srivastava, 2011]**

Weight matrix
$m \times m$

Incidence matrix
$m \times n$

Moore-Penrose
Pseudoinverse of $L$
$n \times n$

# Projection-based approximation

- Johnson- Lindenstrauss Transform:
  - project the system into lower-dymensional space spanned by $\log n/\epsilon^2$ random vectors
  - approximated distances are within $(1+\epsilon)$ factor from exact ones
- Effective resistance $d_{CF}(u, v)$ can be expressed as distances between vectors in $\{W^{1/2}BL^\dagger e_u\}_{u \in V}$ **[Spielman, Srivastava, 2011]**
- Approximation $\{QW^{1/2}BL^\dagger e_u\}_{u \in V}$, $Q$ random projection matrix of size $k \times m$ with elements in $\{0, +\frac{1}{\sqrt{k}}, -\frac{1}{\sqrt{k}}\}$
- Rows of $QW^{1/2}BL^\dagger$: $k$ linear systems:

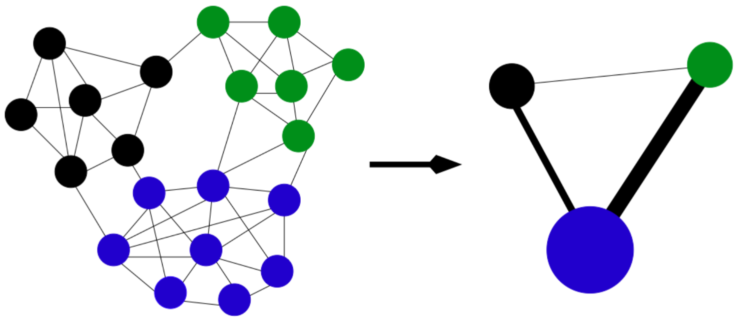
$$Lz_i = \{QW^{1/2}B\}$$

Karlsruhe Institute of Technology

# Implementation

# Laplacian linear systems

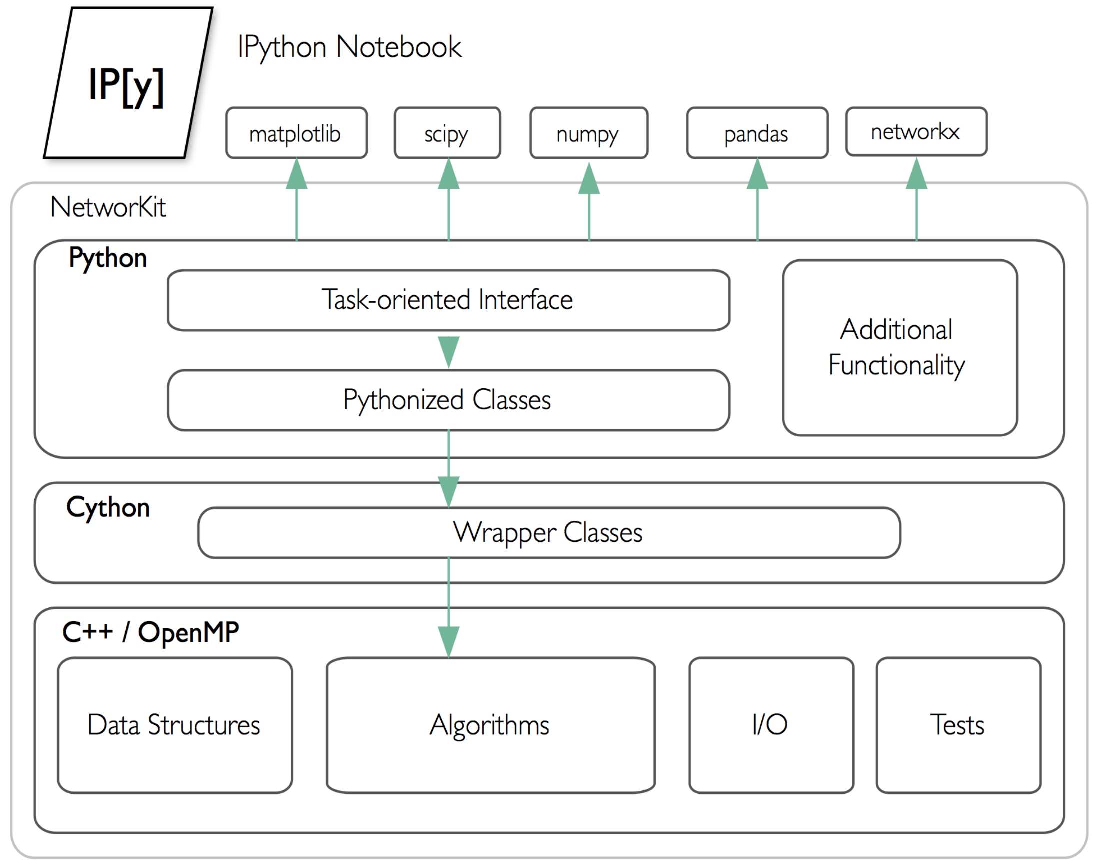
- Laplacian linear systems used to solve many problems in network analysis:
  - Graph partitioning
  - Approx. maximum flow
  - ...
  - Sparsification
  - Graph drawing
- Important to have a fast solver implementation
- LAMG [Livne and Brandt, 2012]:
  - Algebraic multigrid:
  - Iteratively solve coarser systems
  - Prolong solutions to original systems
  - Designed for complex networks



➡ LAMG implementation in NetworKit

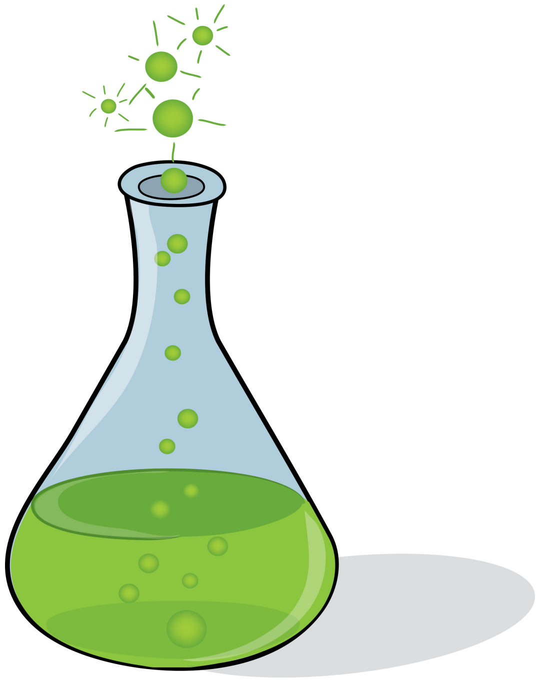
# NetworKit

- a **tool suite of high-performance network analysis algorithms**
  - parallel algorithms
  - approximation algorithms
- **features** include . . .
  - community detection
  - centrality measures
  - graph generators
- **free software**
  - Python package with C++ backend
  - under continuous development
  - download from **http://networkit.iti.kit.edu**



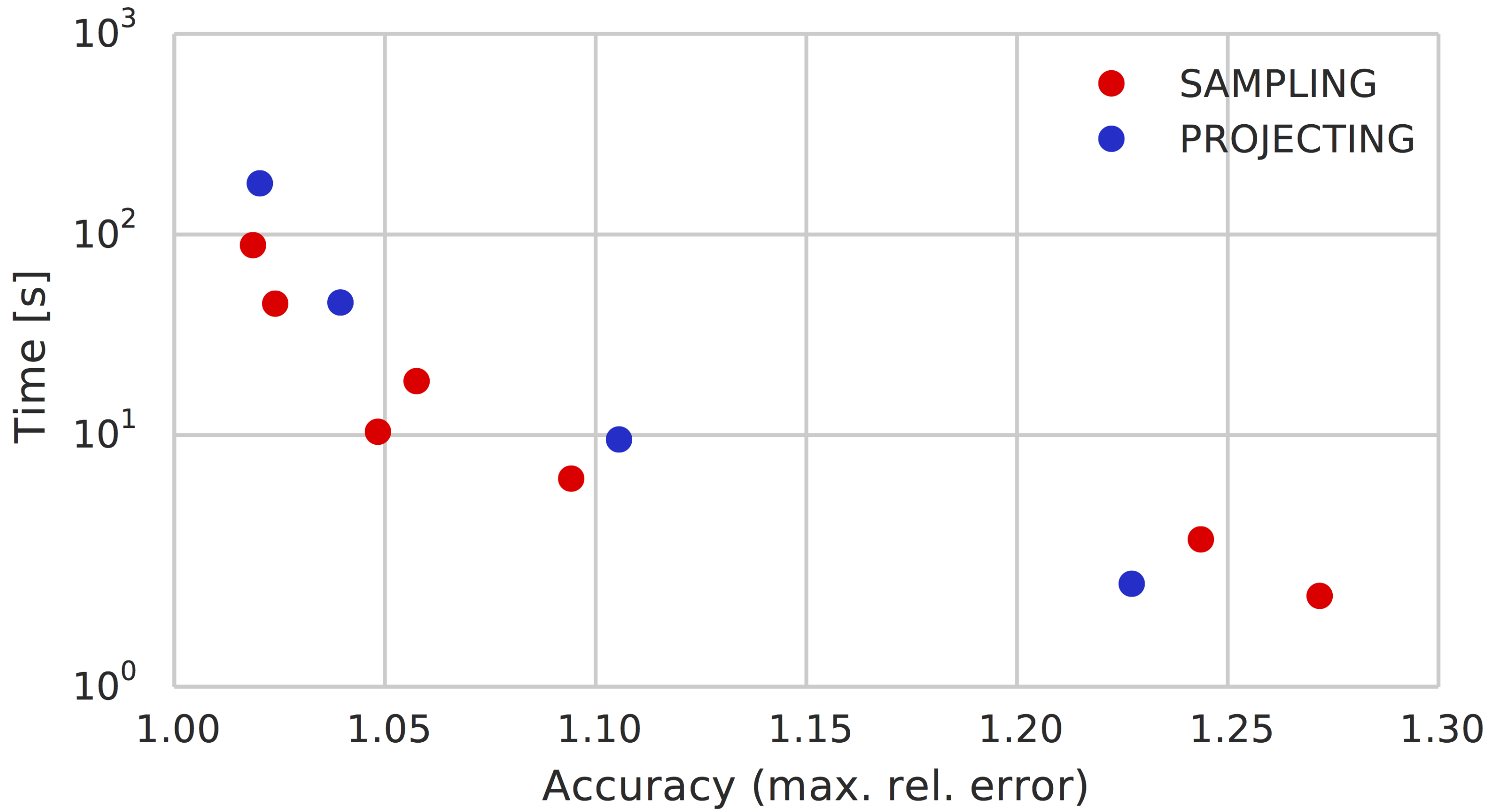

➡ LAMG solver implementation in NetworKit

# Experiments



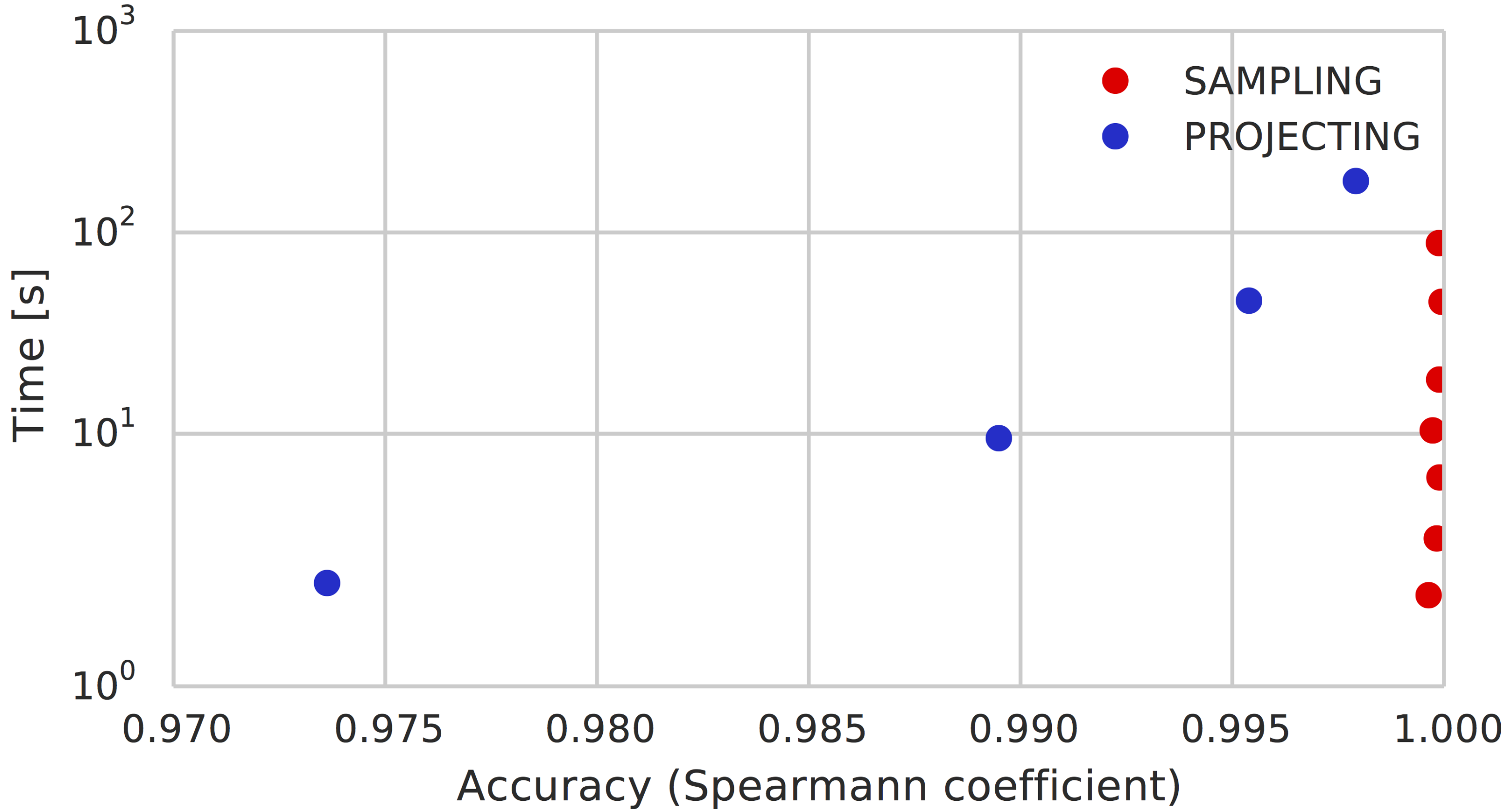
# Approximation algorithms

- Comparison with exact algorithm: networks with up to $10^5$ edges, larger instances up to 56 millions edges
- SAMPLING: $|S| \in \{10, 20, 50, 100, 200, 500\}$
- PROJECTING: $\epsilon = 0.5, 0.2, 0.1, 0.05$



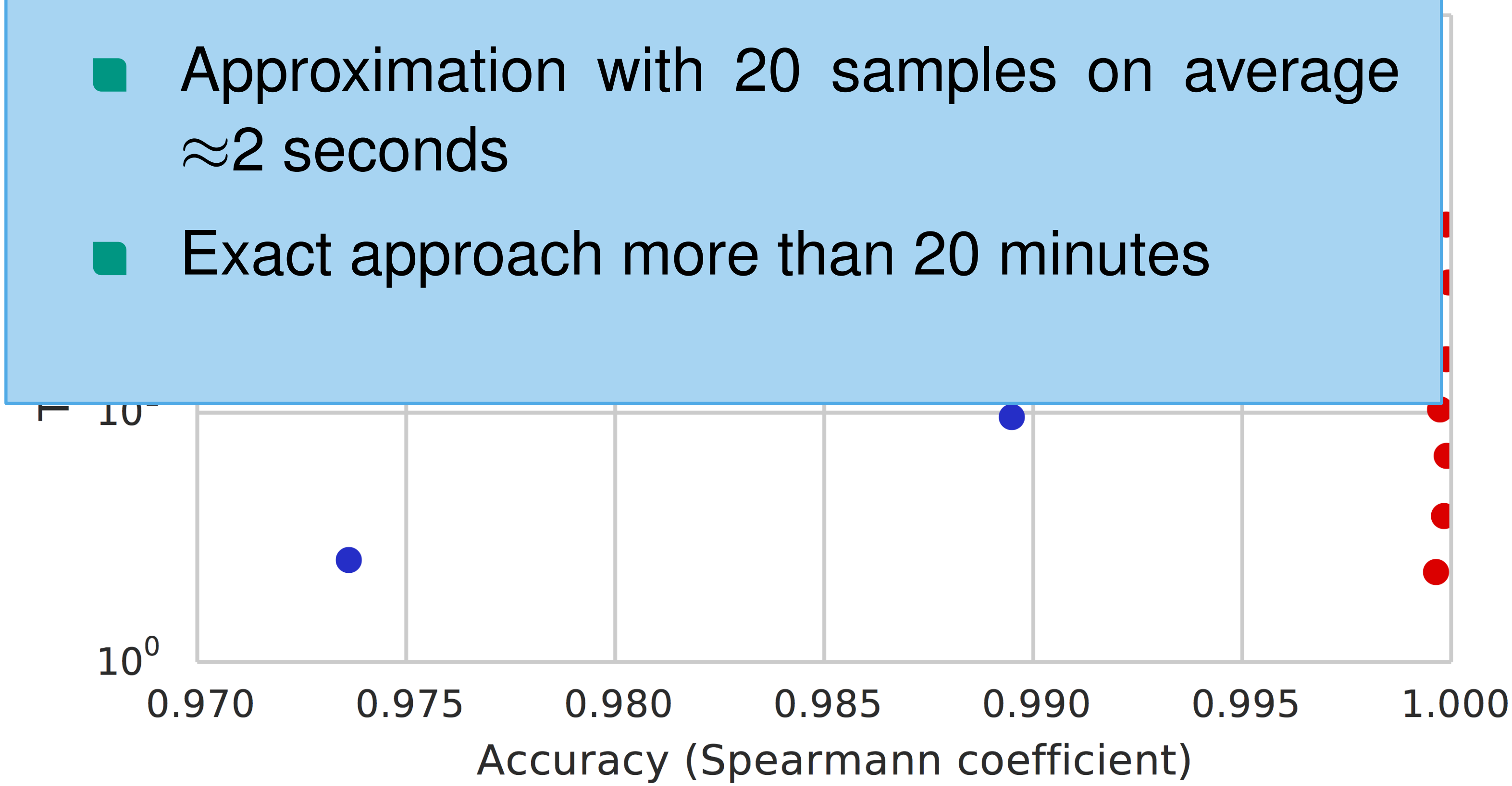
# Approximation algorithms

- Comparison with exact algorithm: networks with up to $10^5$ edges, larger instances up to 56 millions edges
- SAMPLING: $|S| \in \{10, 20, 50, 100, 200, 500\}$
- PROJECTING: $\epsilon = 0.5, 0.2, 0.1, 0.05$



# Approximation algorithms

- Comparison with exact algorithm: networks with up to $10^5$ edges, larger instances up to 56 millions edges
- SAMPLING: $|S| \in \{10, 20, 50, 100, 200, 500\}$
- PROJECTING: $\epsilon = 0.5, 0.2, 0.1, 0.05$



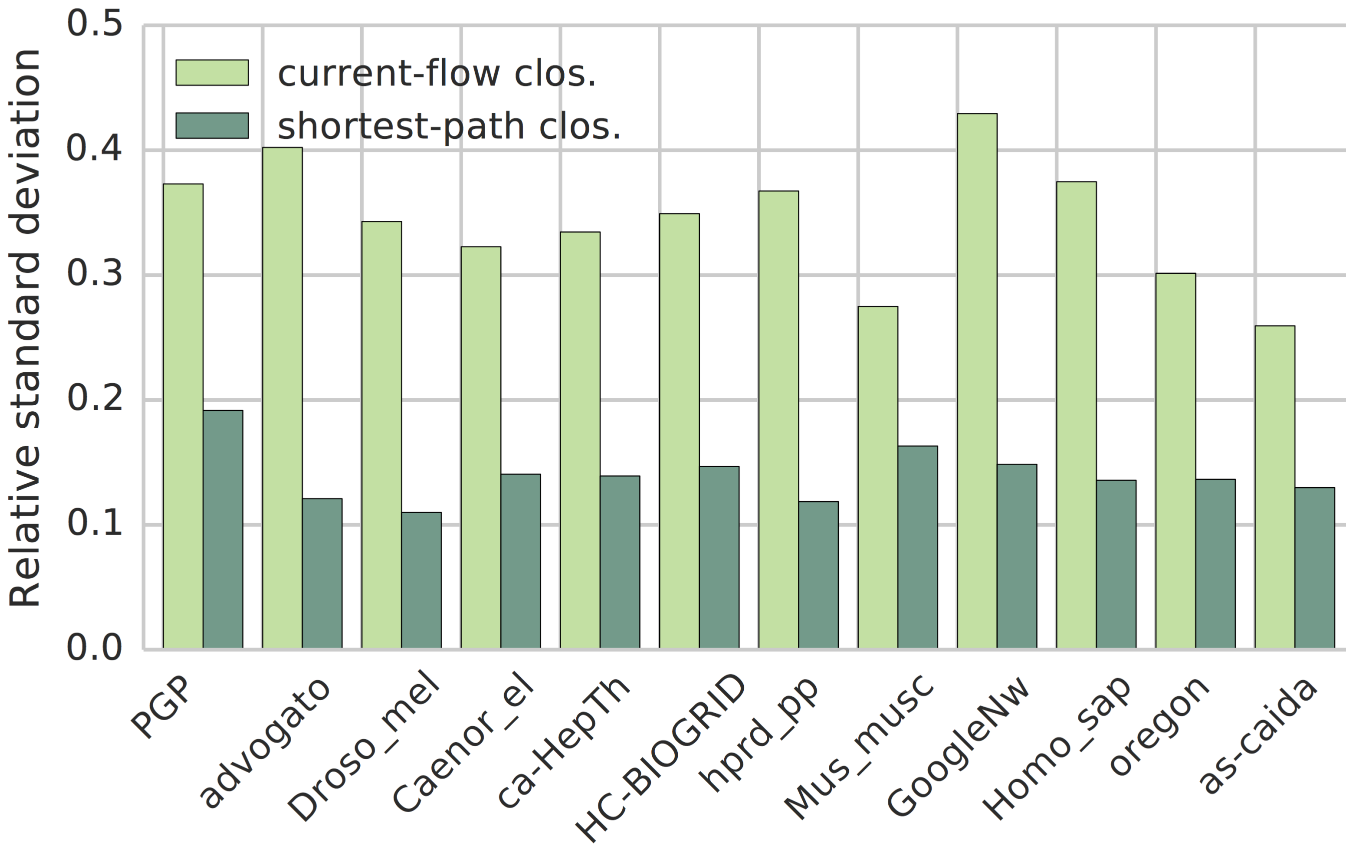

- Approximation with 20 samples on average ≈2 seconds
- Exact approach more than 20 minutes

# Comparison with shortest-path closeness

## Differentiation among different nodes

- Real-world complex networks have small diameters

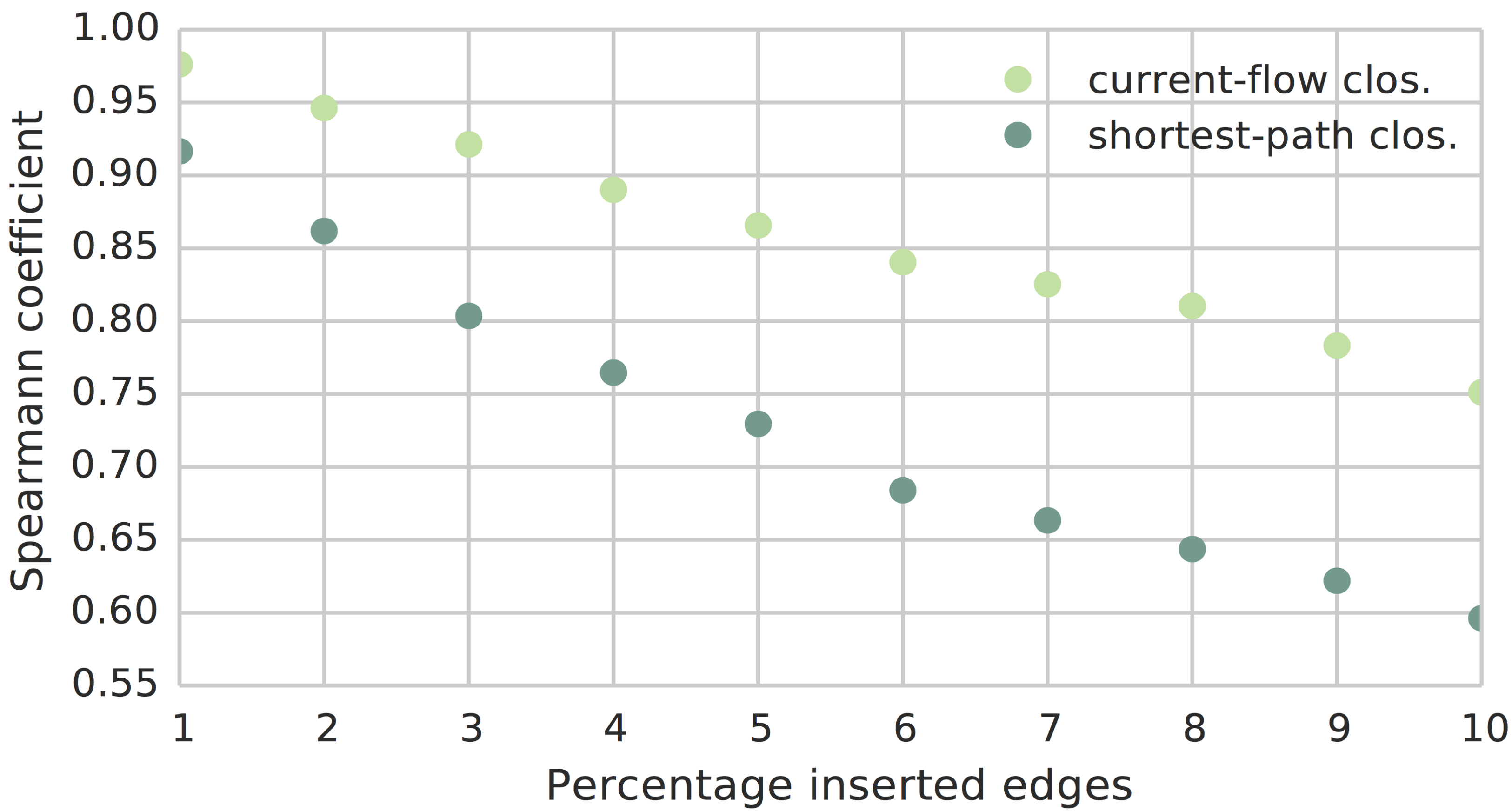
➡ Many nodes have similar shortest-path closeness



# Comparison with shortest-path closeness

## Resilience to noise

- Add new edges to the graph
- Recompute ranking



# Conclusions and future work

- Two approximation algorithms for current-flow closeness of one node
- Current-flow closeness is an interesting alternative to shortest-path closeness

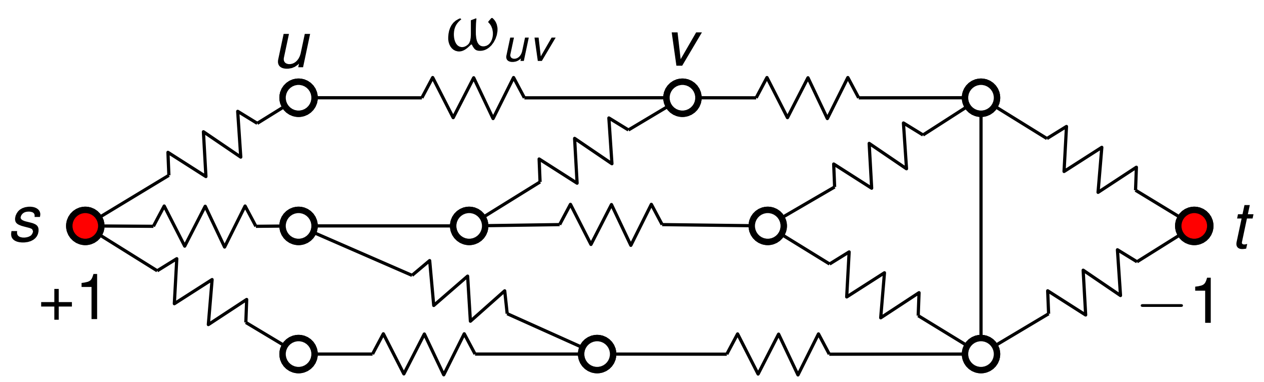
  ⇨ What about electrical betweenness?
- Finding the most central nodes faster?
  (Shortest-path closeness: **[Bergamini et al., ALENEX 2016]**)
- Group centrality

# Conclusions and future work

- Two approximation algorithms for current-flow closeness of one node
- Current-flow closeness is an interesting alternative to shortest-path closeness
  - ⇨ What about electrical betweenness?
- Finding the most central nodes faster?
  (Shortest-path closeness: **[Bergamini et al., ALENEX 2016]**)
- Group centrality

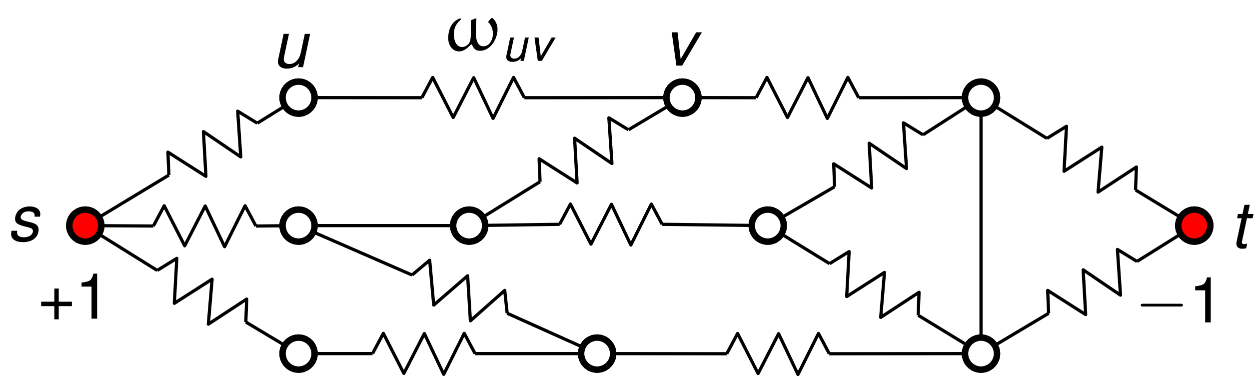
Thank you for your attention!

- Graph as electrical network
- Edge $\{u, v\}$: resistor with conductance $\omega_{uv}$
- Supply $b : V \to \mathbb{R}$
- $b(s) = +1$, $b(t) = -1$ ⇨ current flowing through the network




- Potential $p_{st}(v) \quad \forall v \in V$
- Current $e_{uv}$ flowing through $\{u, v\}$: $(p_{st}(u) - p_{st}(v)) \cdot \omega_{uv}$

- Graph as electrical network
- Edge $\{u, v\}$: resistor with conductance $\omega_{uv}$
- Supply $b : V \rightarrow \mathbb{R}$
- $b(s) = +1$, $b(t) = -1$ ➡ current flowing through the network




Potential can be computed solving the linear system:

$$Lp_{st} = b_{st}$$

where $L := D - A$