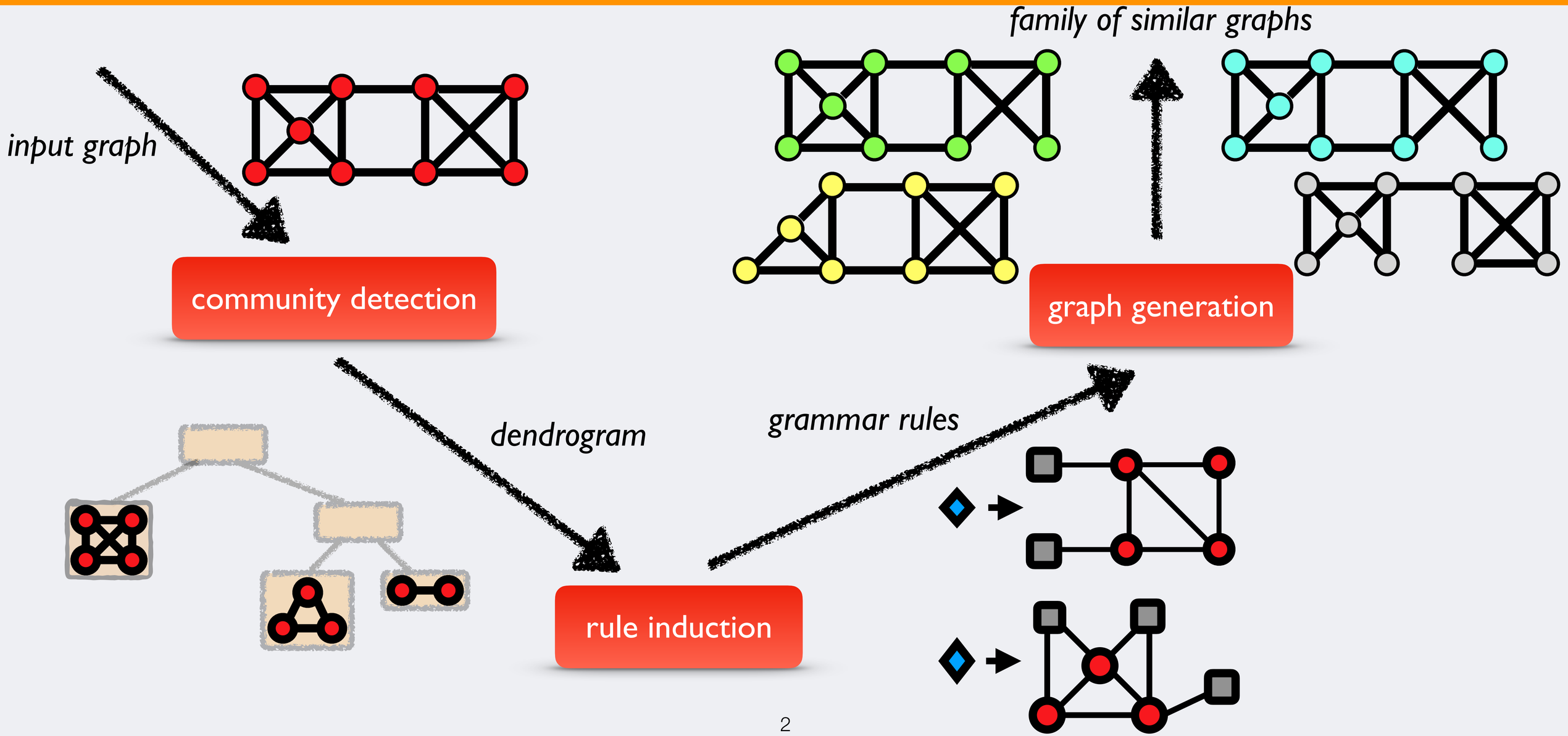
Spectral Community Detection

Initial Implementation

Satyaki Sikdar

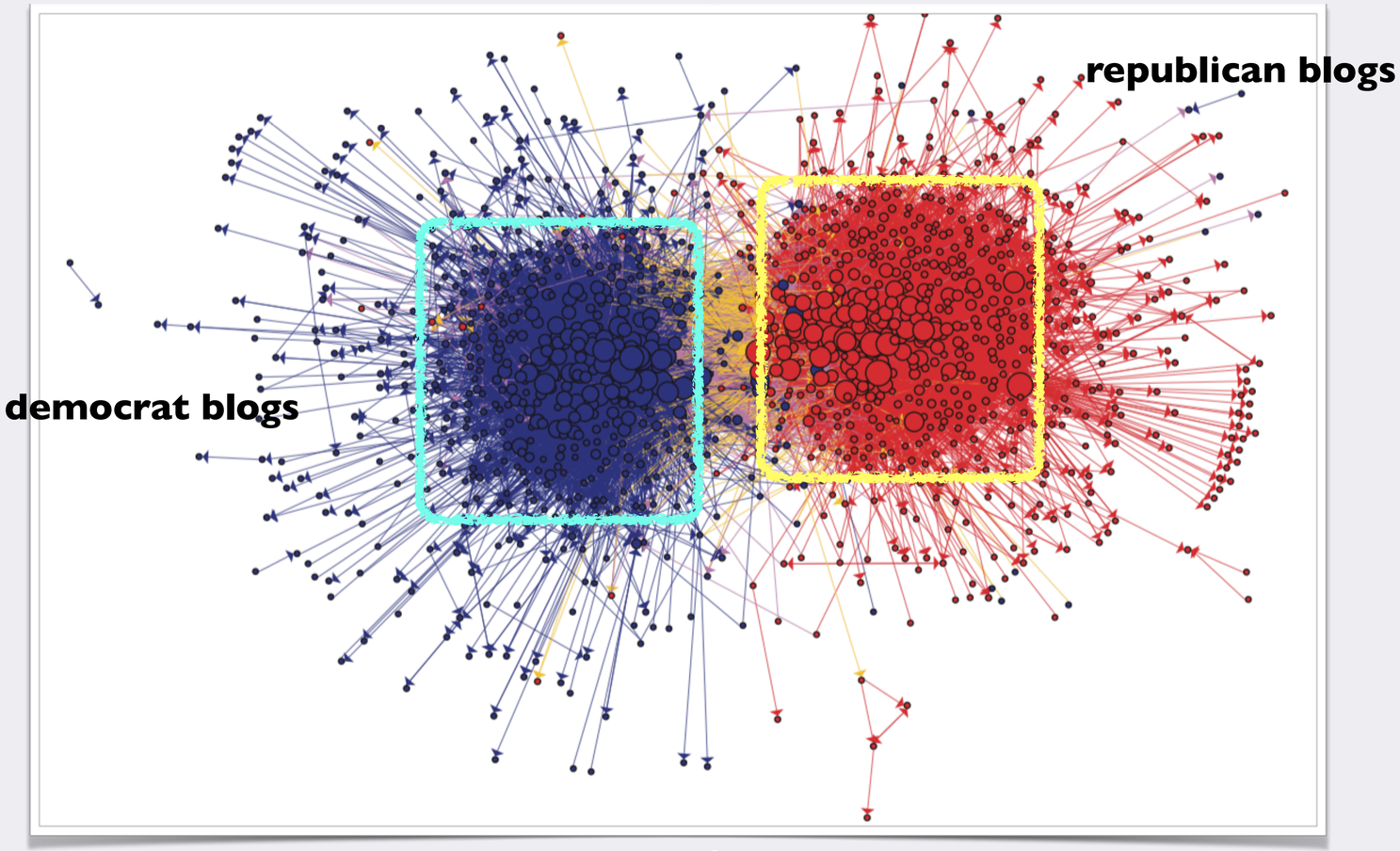
my research pipeline



political homophily

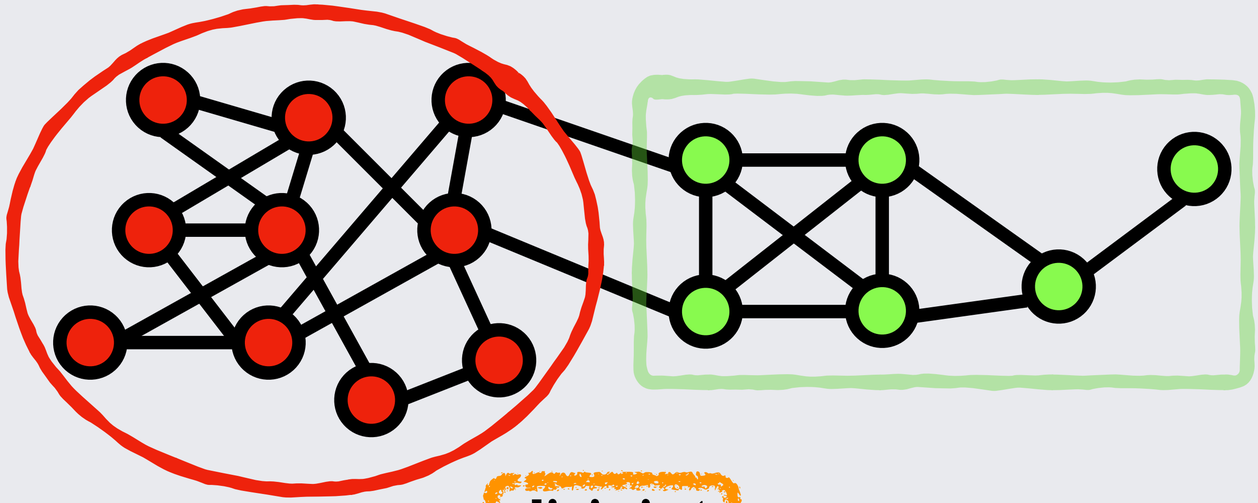
The political blogosphere and the 2004 US election: divided they blog

LA **Adamic**, N Glance - Proceedings of the 3rd international workshop on …, 2005 - dl.acm.org



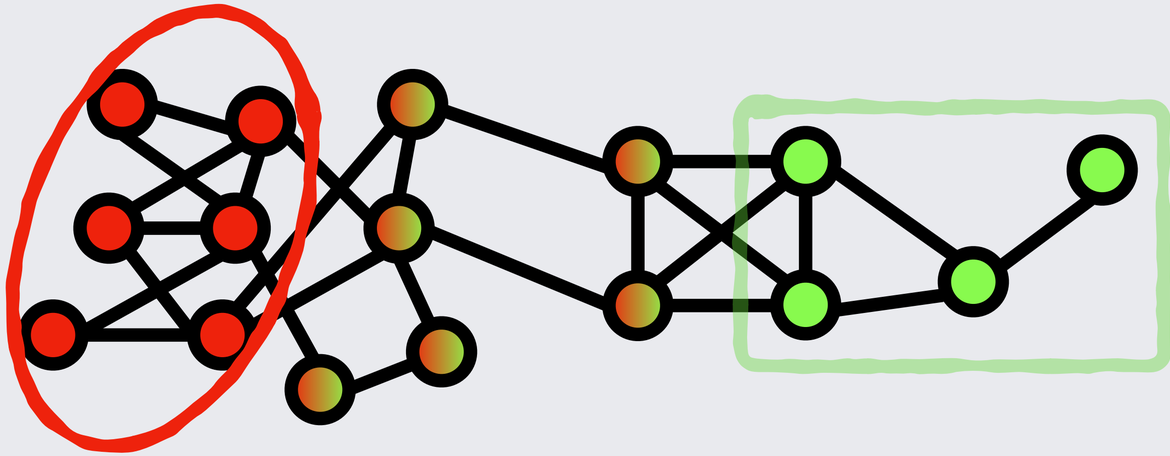
community detection

In a graph $G(V, E)$, find a cover $\mathbb{C} = \{C_1, \cdots, C_k\}$ such that $\bigcup_i C_i = V$



disjoint

$$C_i \cap C_j = \emptyset \quad \forall i, j$$



overlapping

$$C_i \cap C_j \neq \emptyset \quad \exists\ i, j$$

spectral clustering algorithms

bipartitions

New **spectral** methods for ratio cut partitioning and **clustering**
L **Hagen**, AB Kahng - … transactions on computer-aided design of …, 1992 - ieeexplore.ieee.org

- clustering based on median / fixed value

Normalized cuts and image segmentation
J **Shi**, J **Malik** - IEEE Transactions on pattern analysis and …, 2000 - ieeexplore.ieee.org

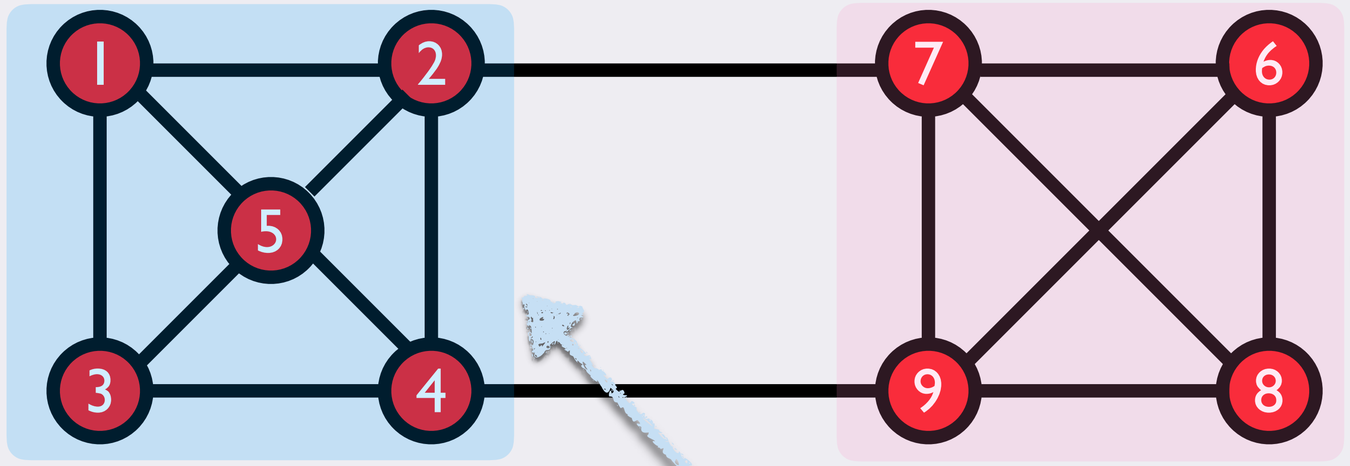
clustering multiple eigenvectors

[PDF] **On spectral clustering**: Analysis and an algorithm
AY Ng, MI Jordan, Y Weiss - Advances in neural information …, 2002 - papers.nips.cc

- spectral embedding of nodes and then k-means

$$L = D - A$$

$$Lx = \lambda x$$



L	1	2	3	4	5	6	7	8	9
1	3	-1	-1		-1				
2	-1	4		-1	-1		-1		
3	-1		3	-1	-1				
4		-1	-1	4	-1				-1
5	-1	-1	-1	-1	4				
6						3	-1	-1	-1
7		-1				-1	4	-1	-1
8						-1	-1	3	-1
9				-1		-1	-1	-1	4

sorted eigenvalues

$$\begin{pmatrix} 9.540\,\mathrm{e}-17 \\ 6.498\,\mathrm{e}-01 \\ 3.198 \\ 3.326 \\ 4 \\ 4.554 \\ 4.641 \\ 5.382 \\ 6.246 \end{pmatrix}$$

fiedler vector

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{array} \begin{pmatrix} -0.378 \\ -0.178 \\ -0.378 \\ -0.332 \\ -0.178 \\ 0.291 \\ 0.291 \\ 0.431 \\ 0.431 \end{pmatrix}$$

bipartition pseudocode

Algorithm 1 Approximate minimum cut of a connected graph G for a given threshold r

```
1:  procedure approx_min_cut(G(V, E), r)
2:      clusters ← ∅
3:      if G has fewer than 2 nodes then
4:          clusters ← V
5:      else
6:          fiedler ← Fiedler vector of G
7:          p1 ← nodes ids with value less than r in fiedler
8:          p2 ← rest of the nodes in G
9:          clusters ← clusters ∪ { p1}
10:         clusters ← clusters ∪ { p2}
11:     return clusters
```

$\mathcal{O}(|V| + |E|)$

Krishnan, S., & Goldberg, K. (2015). The minimum conductance dissimilarity cut (MCDC) algorithm to increase novelty and diversity of recommendations. Working paper, Industrial Engineering and Operations Research Department, University of California, Berkeley, CA.

python implementation

```
import networkx as nx

def approx_min_cut(G, r):
    clusters = []
    if G.order() < 2:
        clusters = list(G.nodes())
    else:
        fiedler_vec = nx.fiedler_vector(G, method='lanczos')
        p1, p2 = set(), set()
        for node_id, fiedler_val in zip(G.nodes(), fiedler_vec):
            if fiedler_val < r:
                p1.add(node_id)
            else:
                p2.add(node_id)
        clusters.append(p1)
        clusters.append(p2)
    return clusters
```

spectral clustering algorithms

bipartitions

New **spectral** methods for ratio cut partitioning and **clustering**
L **Hagen**, AB Kahng - … transactions on computer-aided design of …, 1992 - ieeexplore.ieee.org

- clustering based on median / fixed value

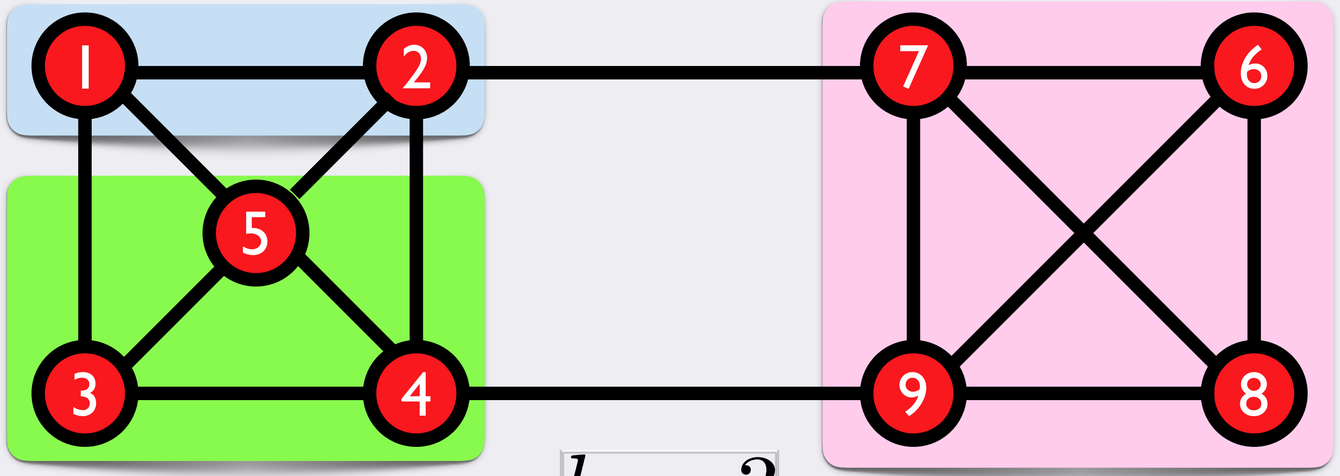
Normalized cuts and image segmentation
J **Shi**, J **Malik** - IEEE Transactions on pattern analysis and …, 2000 - ieeexplore.ieee.org

clustering multiple eigenvectors

[PDF] **On spectral clustering**: Analysis and an algorithm
AY Ng, MI Jordan, Y Weiss - Advances in neural information …, 2002 - papers.nips.cc

- spectral embedding of nodes and then k-means

$$L = D - A$$



$k = 3$

L	1	2	3	4	5	6	7	8	9
1	3	-1	-1		-1				
2	-1	4		-1	-1		-1		
3	-1		3	-1	-1				
4		-1	-1	4	-1				-1
5	-1	-1	-1	-1	4				
6						3	-1	-1	-1
7		-1				-1	4	-1	-1
8						-1	-1	3	-1
9				-1		-1	-1	-1	4

sorted eigenvalues

$$\begin{pmatrix} 9.540\,\mathrm{e}-17 \\ 6.498\,\mathrm{e}-01 \\ 3.198 \\ 3.326 \\ 4 \\ 4.554 \\ 4.641 \\ 5.382 \\ 6.246 \end{pmatrix}$$

k-smallest eigenvector values

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{array} \begin{pmatrix} -0.378 & 0.521 & -0.428 \\ -0.178 & 0.418 & 0.463 \\ -0.378 & -0.521 & -0.428 \\ -0.332 & 0 & 0.104 \\ -0.178 & -0.418 & 0.463 \\ 0.291 & 0.232 & 0.172 \\ 0.291 & -0.232 & 0.172 \\ 0.431 & 0 & -0.26 \\ 0.431 & 0 & -0.26 \end{pmatrix}$$

pseudocode

Algorithm 2 k-way spectral partitioning of a connected graph G

```
1:  procedure k_way_spectral(G(V, E), k)
2:      clusters ← ∅
3:      if G has fewer than k nodes then
4:          clusters ← V
5:      else
6:          L ← Laplacian matrix of G
7:          evecs ← k-smallest non-trivial eigenvectors of L
8:          Normalize each row of evecs by its norm
9:          Run K-means clustering on evecs to find k clusters ℂ = {C_1, ··· , C_k}
10:         clusters ← ℂ
11:     return clusters
```

python implementation

```
import networkx as nx; import numpy as np
import scipy.sparse.linalg; from sklearn.cluster import KMeans

def k_way_spectral(G, k):
    clusters = []
    if G.order() < k:
        clusters = list(G.nodes())
    else:
        L = nx.laplacian_matrix(G)
        # compute the first k + 1 eigenvectors
        _, eigenvecs = scipy.sparse.linalg.eigsh(L, k=k+1, which='SM')
        eigenvecs = eigenvecs[:, 1:] # discard the first trivial eigenvector
        # normalize each row by its norm
        eigenvecs = np.apply_along_axis(lambda x: x / np.linalg.norm(x),
                                        axis=1, arr=eigenvecs)
        # run K-means
        kmeans = KMeans(n_clusters=k).fit(eigenvecs)
        cluster_labels = kmeans.labels_
        clusters = [[] for _ in range(max(cluster_labels) + 1)]
        for node_id, cluster_id in zip(G.nodes(), cluster_labels):
            clusters[cluster_id].append(node_id)
    return clusters
```

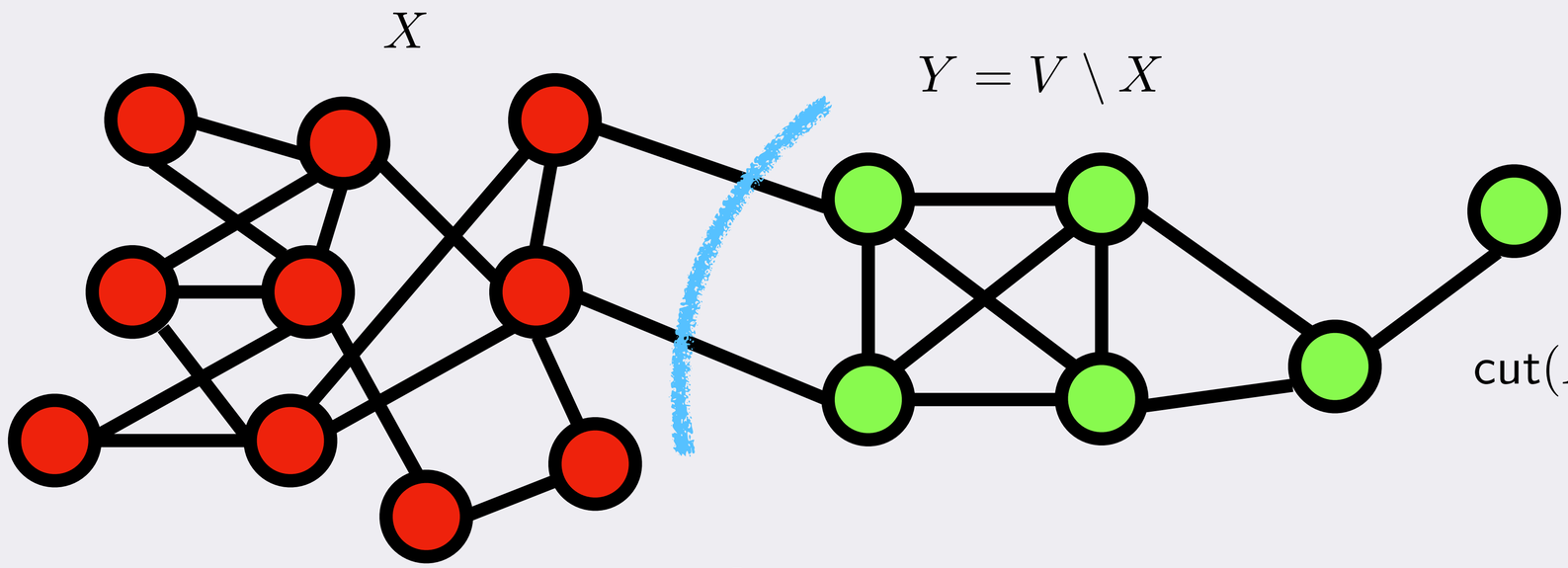
quality measures - conductance

degree

$$d_i = \sum_j \mathbf{A}_{ij}$$

volume

$$\text{vol}(X) = \sum_{j \in X} d_i$$



cut

$$\text{cut}(X, Y) = \sum_{i \in X,\, j \in Y} \mathbf{A}_{ij}$$

conductance

$$\phi(X) = \frac{\text{cut}(X, V \setminus X)}{\min\{\text{vol}(X), \text{vol}(V \setminus X)\}}$$

lower is better!

Kannan, R., Vempala, S., & Vetta, A. (2004). On clusterings: Good, bad and spectral. Journal of the ACM (JACM), 51(3), 497-515

quality measures - modularity

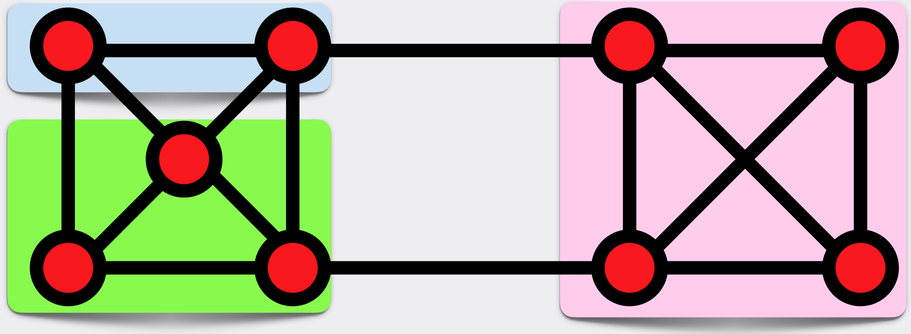
$$Q(\mathbb{C}) = \sum_{C \in \mathbb{C}} \left[\frac{l_C}{|E|} - \left(\frac{k_C}{2|E|} \right)^2 \right]$$

set of clusters $\mathbb{C} = \{C_1, C_2, \cdots, C_k\}$

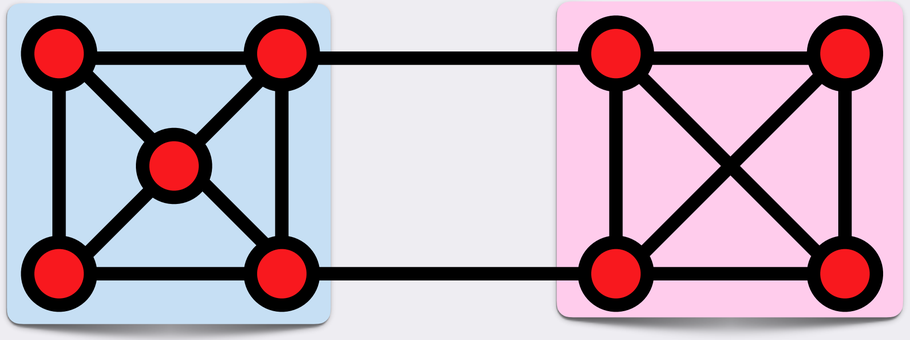
number of edges inside cluster C

sum of degrees of nodes in cluster C

higher is better!



$Q = 0.2675$



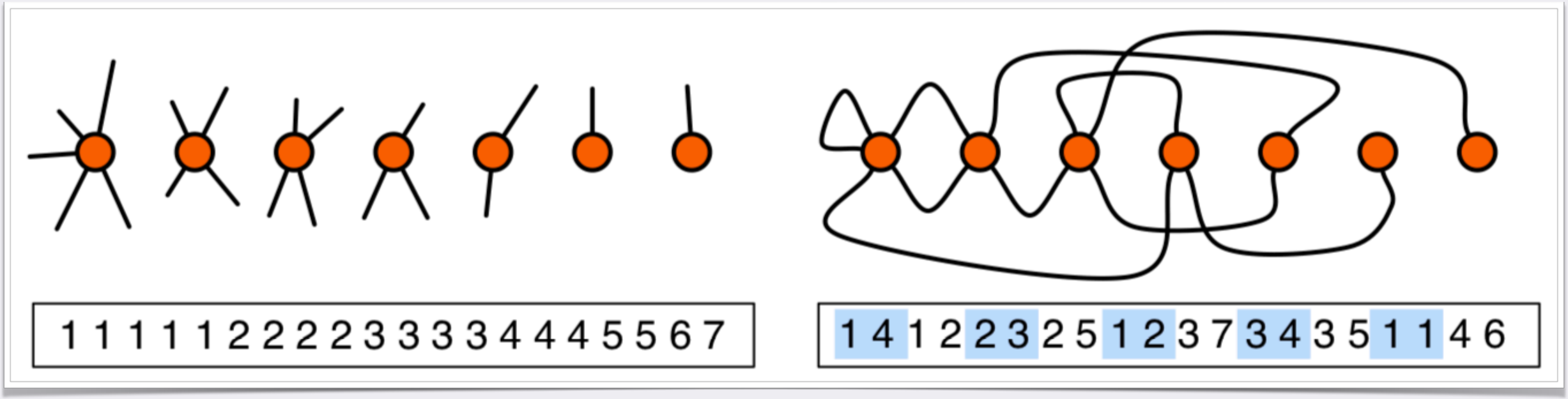
$Q = 0.3671$

Newman, M. E. (2006). Modularity and community structure in networks. ***Proceedings of the national academy of sciences, 103(23), 8577-8582.***

configuration model

Generates a random graph that follows a given degree sequence

$$\vec{k} = \{k_1, k_2, \cdots, k_l\}$$



http://tuvalu.santafe.edu/~aaronc/courses/5352/fall2013/csci5352_2013_L11.pdf

Chung, F., & Lu, L. (2002). Connected components in random graphs with given expected degree sequences. *Annals of combinatorics*, *6*(2), 125-145.

LFR benchmark

generates graphs with tunable degree and community size distributions

1. number of nodes n

2. average degree $\langle k \rangle$

3. power-law exponents for degree γ and community sizes β

4. mixing parameter μ

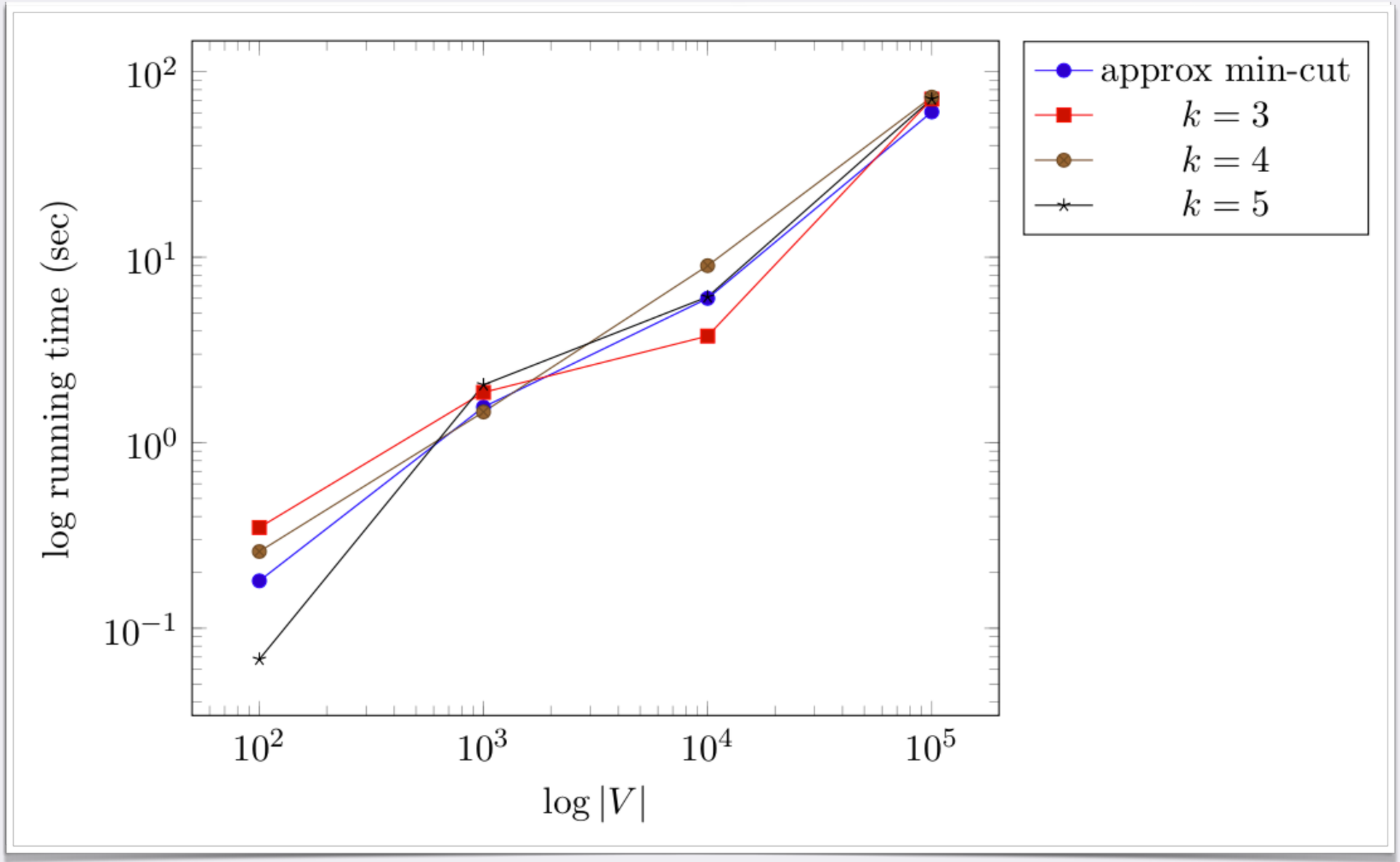
Lancichinetti, A., Fortunato, S., & Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Physical review E*, *78*(4), 046110.

scaling results

Average running times in seconds across 5 runs on graphs generated with the LFR benchmark with $\langle k \rangle = 16$, $\gamma = -2$, $\beta = -1$, $\mu = 0.1$

			k-way spectral		
$\lvert V \rvert$	$\lvert E \rvert$	approx min-cut	$k = 3$	$k = 4$	$k = 5$
100	795	0.018	0.349	0.259	0.068
1,000	7,692	1.556	1.867	1.464	2.051
10,000	76,325	5.997	3.757	8.998	6.109
100,000	765,073	60.654	71.142	72.926	71.33

running time plot



parallelization plans

Parallel Spectral Clustering

Yangqiu Song[1,4], Wen-Yen Chen[2,4], Hongjie Bai[4],
Chih-Jen Lin[3,4], and Edward Y. Chang[4]

Parallel Spectral Graph Partitioning

Maxim Naumov and Timothy Moon
NVIDIA, 2701 San Tomas Expressway, Santa Clara, CA 95050

numpy and scipy already use BLAS

the google research paper uses BLAS with MPI

A High Performance Implementation of Spectral Clustering on CPU-GPU Platforms

Yu Jin
Institute for Advanced Computer Studies
Department of Electrical and Computer Engineering
University of Maryland, College Park, USA
Email: yuj@umd.edu

Joseph F. JaJa
Institute for Advanced Computer Studies
Department of Electrical and Computer Engineering
University of Maryland, College Park, USA
Email: joseph@umiacs.umd.edu

use a parallel version of k-means - scikit-learn has that too! 😍

using a JIT compiler

thanks!