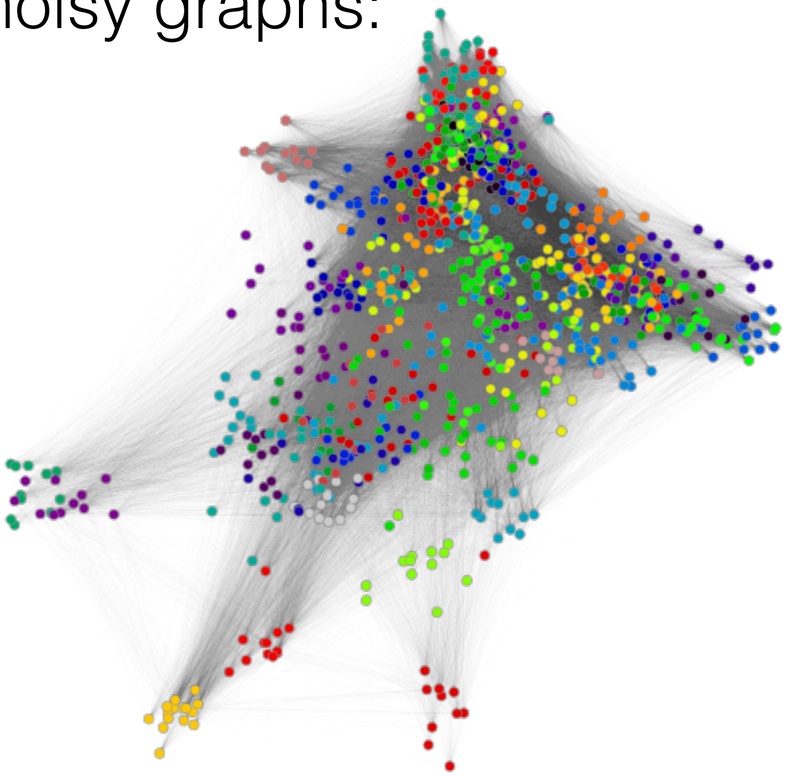
Recent advances in local graph clustering and the transition to global analysis

Kimon Fountoulakis @CS UWaterloo
02/07/2020
Workshop: From Local to Global Information

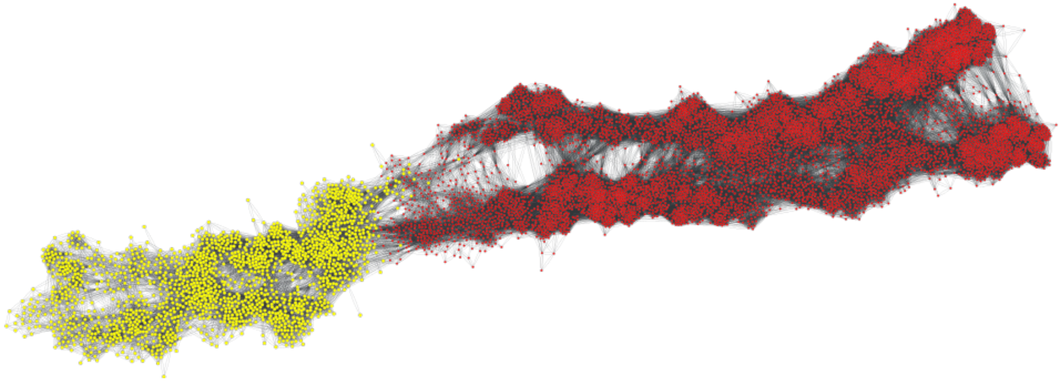
Motivation: detection of small clusters in large and noisy graphs

- Real large-scale graphs have rich local structure
- We often have to detect small clusters in large and noisy graphs:



protein-protein interaction graph,
color denotes similar functionality

Rather than partitioning graphs with nice structure



US-Senate graph,
nice bi-partition in year 1865 around the end of the American civil ward

Our goals

Large scale data with multiple noisy small-scale and meso-scale clusters determine the need for

- new methods that are able probe graphs with billions of nodes and edges,

- the running time of the new methods should depend on the size of the output instead of the size of the whole graph,

- the new methods should be supported by worst- and average-case theoretical guarantees.

Existing and new local graph clustering methods

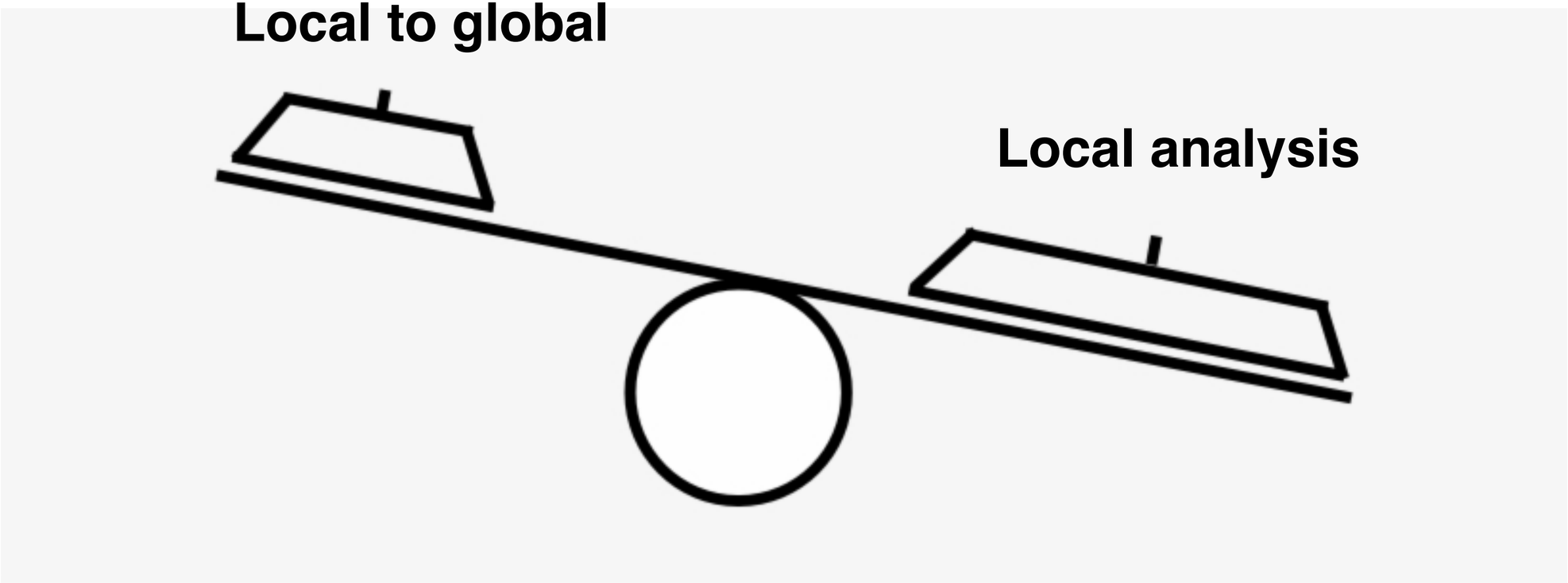
The vast majority of methods perform some sort of linear diffusion, i.e., PageRank. We need models that are better than simply averaging of probabilities.

- As a warm-up: non-linear PageRank.

- Non-linear combinatorial diffusions.

- Non-linear diffusions which balance between spectral and combinatorial diffusions.

Current local and global developments for local graph clustering methods



About this talk

-I will mostly discuss methods, I will demonstrate theoretical results and I will present experiments that promote understanding of the methods within the available time.

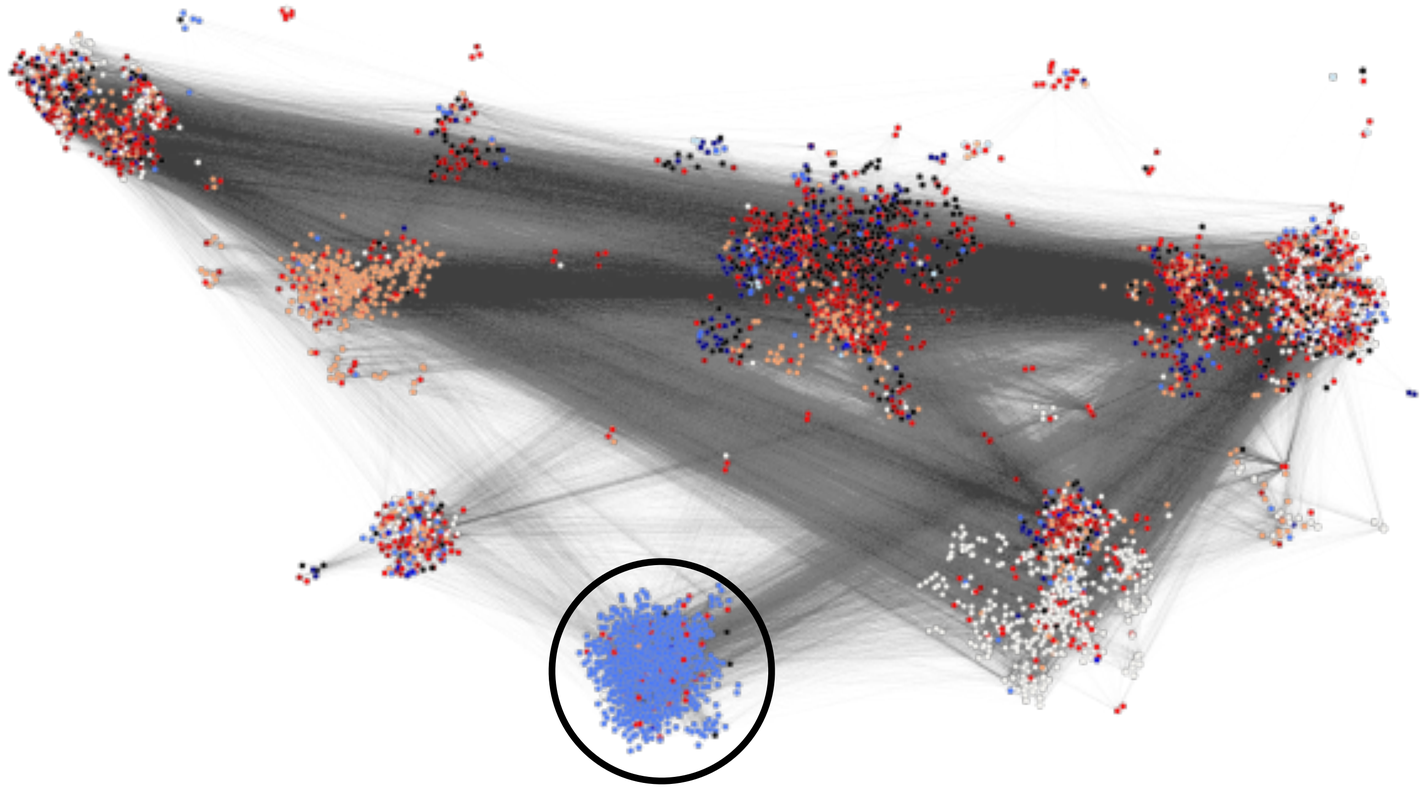
-For extensive experiments on real-data please check the cited papers. We literally have performed hundreds of experiments for measuring performance of local graph clustering methods.

Local Graph Clustering

The local graph clustering problem?

- Definition: find set of nodes A given a seed node in set B
  - Set A has good precision/recall w.r.t set B
  - The running time depends on A instead of the whole graph

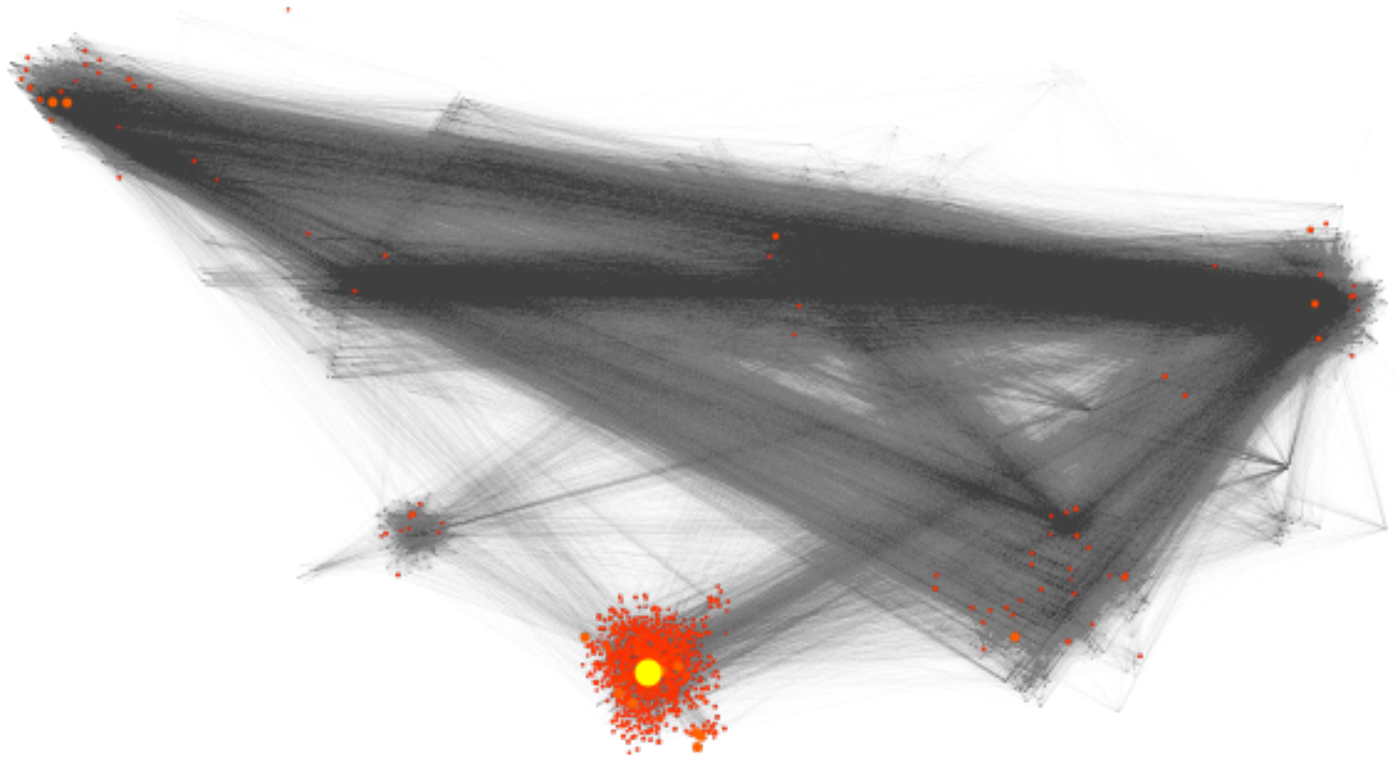
Facebook Johns Hopkins social network: color denotes class year



Students of year 2009

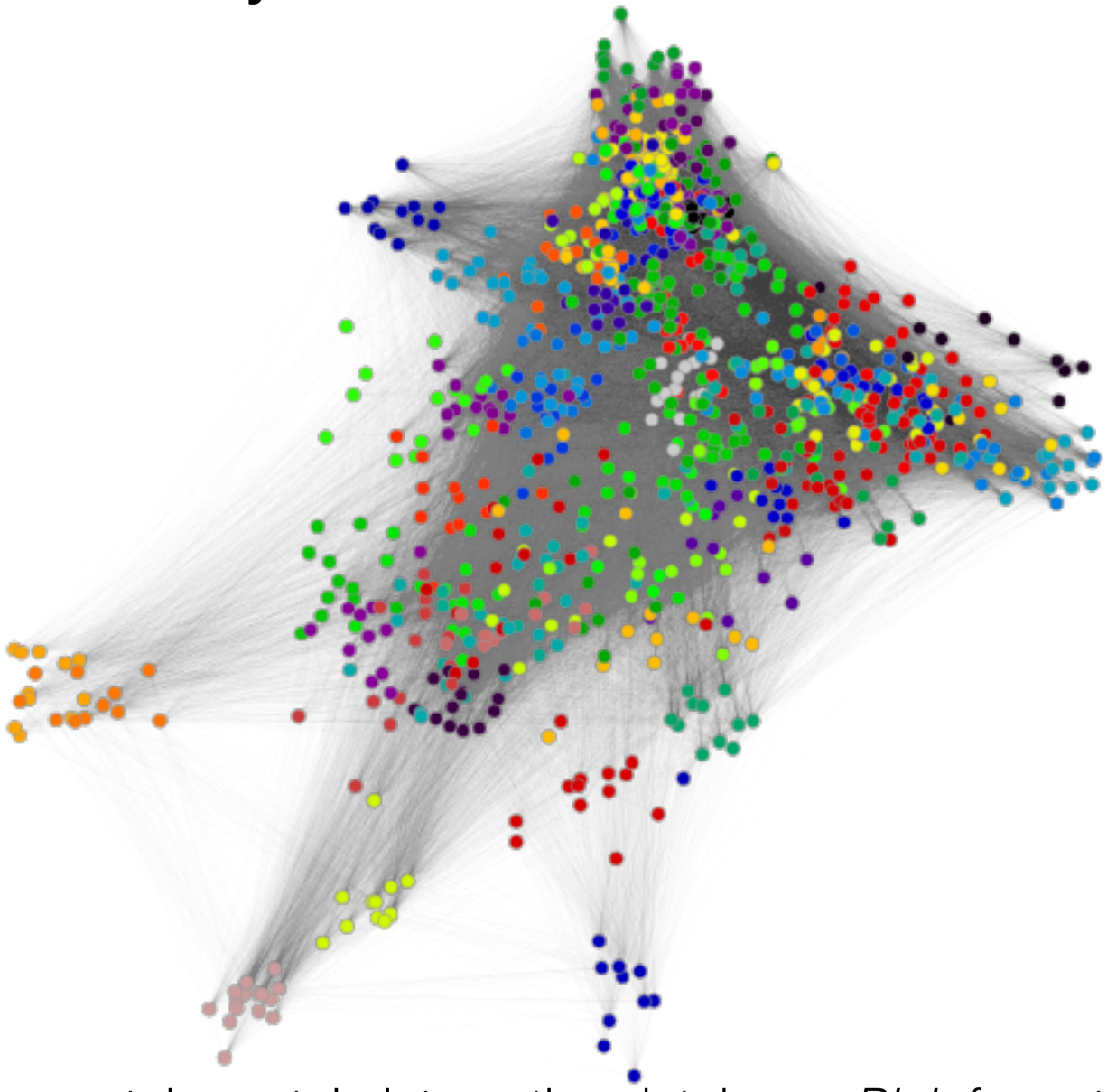
Data: Facebook Johns Hopkins, A. L. Traud, P. J. Mucha and M. A. Porter, Physica A, 391(16), 2012

Local graph clustering: example



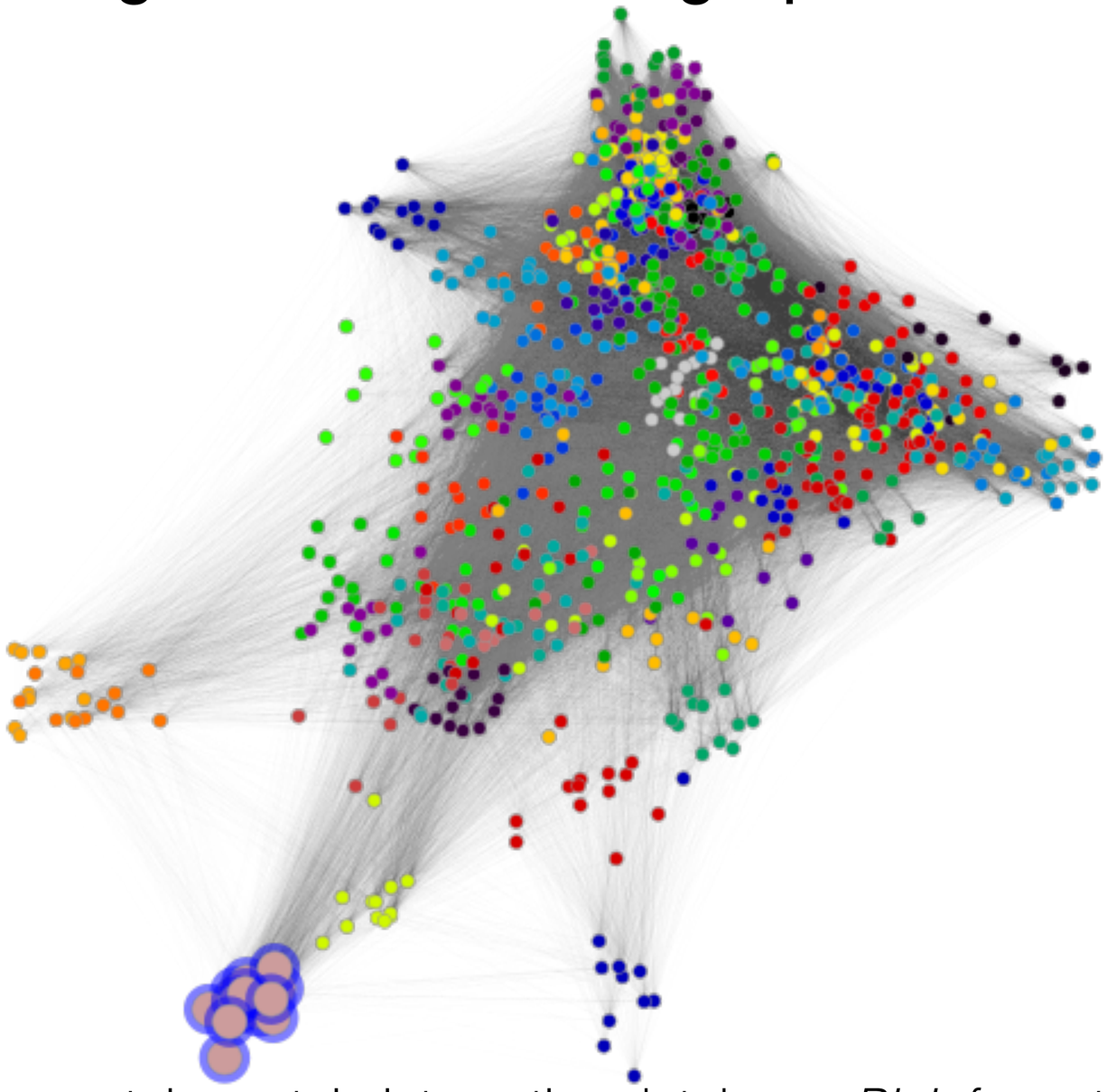
Data: Facebook Johns Hopkins, A. L. Traud, P. J. Mucha and M. A. Porter, Physica A, 391(16), 2012

Protein structure similarity: color denotes similar function



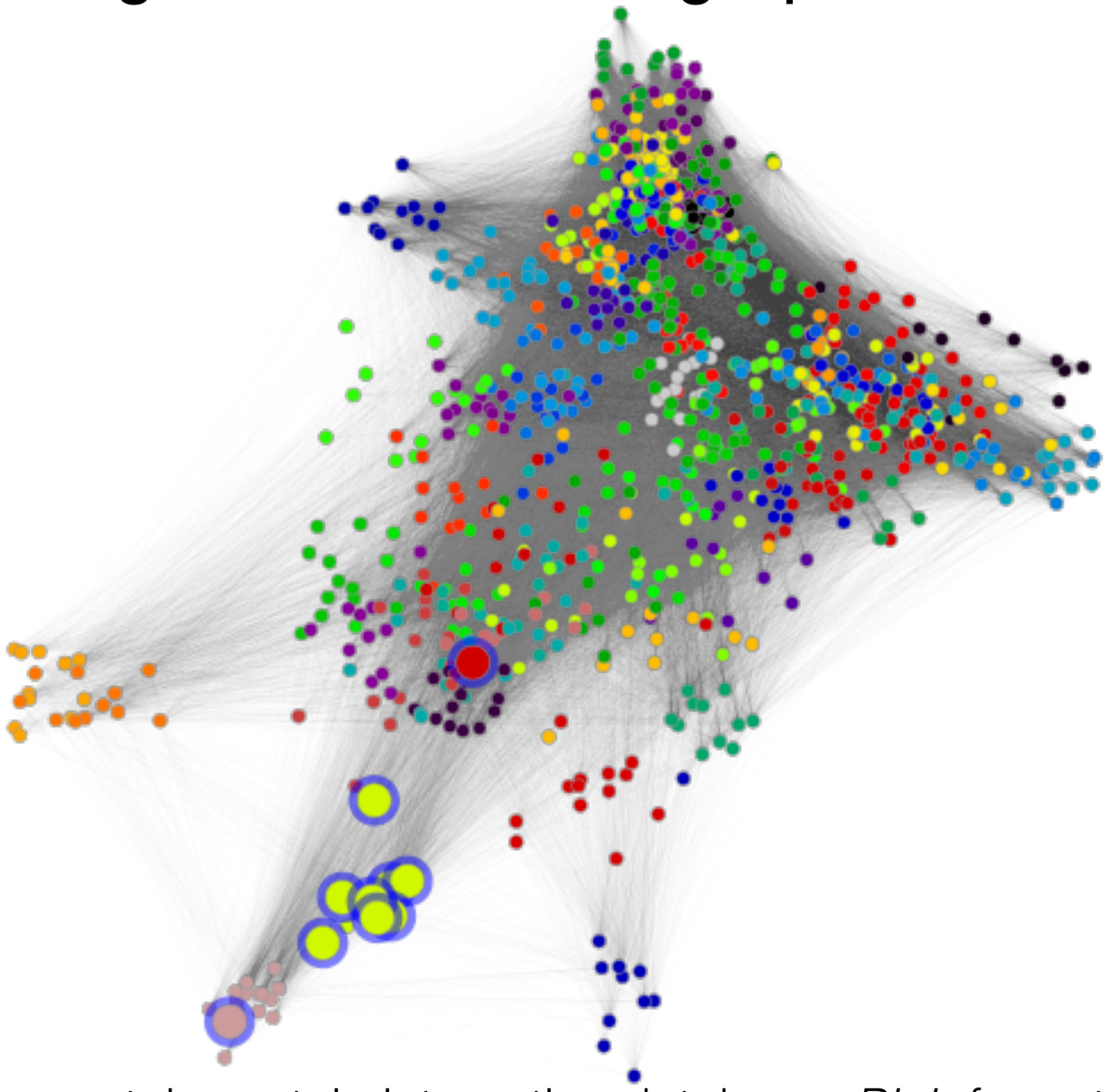
Data: The MIPS mammalian protein-protein interaction database. *Bioinformatics, 21(6):832-834*, 2005

Local graph clustering finds 2% of the graph



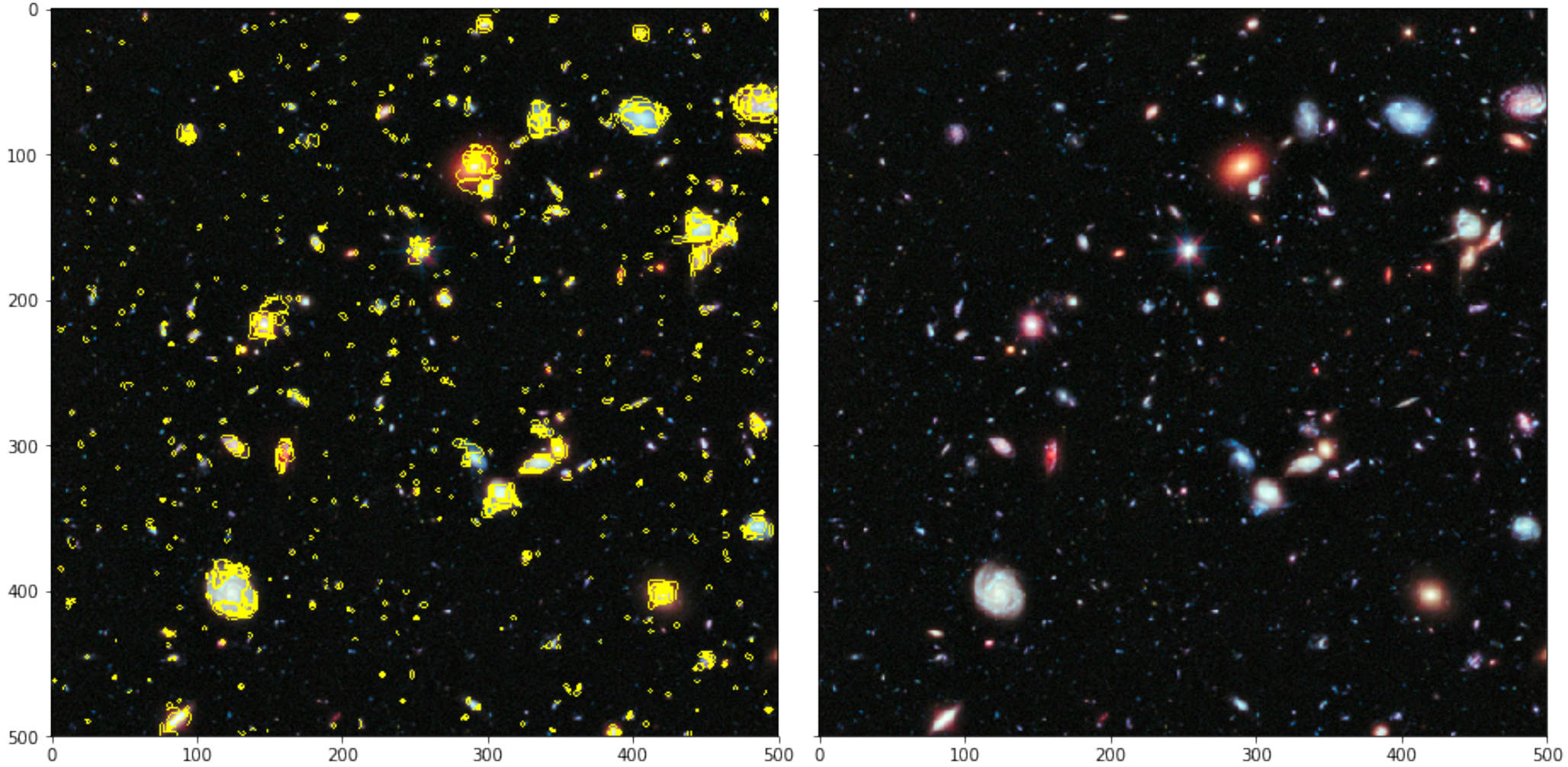
Data: The MIPS mammalian protein-protein interaction database. *Bioinformatics, 21(6):832-834*, 2005

Local graph clustering finds 1% of the graph



Data: The MIPS mammalian protein-protein interaction database. *Bioinformatics, 21(6):832-834*, 2005

Or we might want to detect galaxies



Warm-up: non-linear PageRank

Some definitions

- Graph: $G = (\underbrace{V}_{nodes}, \underbrace{E}_{edges}), |V| = n, |E| = m$

- n x n adjacency matrix: A

- An element of A is equal to 1 if two nodes are connected

Some definitions

- Degree matrix: $D = diag(A1_n)$, 1_n is a vector of all ones.
- Each element of D shows the number of neighbors of a node
- Random walk matrix: AD^{-1}
- Lazy random walk matrix: $W = \frac{1}{2}\left(I + AD^{-1}\right)$
- Graph Laplacian: $L = D - A$

Linear diffusion: personalized PageRank

- Let $\alpha \in (0,1)$ be the teleportation parameter

- Consider a diffusion process where we perform lazy random walk with probability $1-\alpha$, and jump to a given seed node with probability α:

$$\alpha s 1_n^T + (1-\alpha)W$$

- where s is an indicator vector of the seed node and alpha is the teleportation parameter.

- **Simple idea:** use a random walk from a seed node. The nodes with the highest probability after k steps consist a cluster.

Let’s get rid off the tail

- For the stationary personalized PageRank vector most of the probability mass is concentrated around the seed node.

- This means that the ordered personalized PageRank vector has long tail for nodes far away from the seed node.

- We can efficiently cut the tale using l1-regularized PageRank without even having to compute the long tail.

Non-linear PageRank diffusion

- Instead of using power method to compute the PageRank vector, we can perform a non-linear power method where we do a random walk step first and then threshold small values to zero.

$$p_{k+1} = \text{prox}_{\rho\alpha d\|\cdot\|_1}(\quad \underbrace{(1-\alpha)Wp_k + \alpha s}_{\text{random walk step}} \quad)$$

- where prox operator reduces components smaller than $\rho\alpha d$ to zero.

$$\text{prox}_{\rho\alpha d\|\cdot\|_1}(x) = \begin{cases} x - \rho\alpha d & \text{if } x \geq \rho\alpha d \\ 0 & \text{otherwise} \end{cases}$$

Far stretched relation to graph neural networks

Non-linear PageRank

$$p_{k+1} = \text{prox}_{\rho\alpha d\|\cdot\|_1}\Big(\underbrace{(1-\alpha)Wp_k + \alpha s}_{\text{random walk step}} \Big)$$

Graph Neural Network Layer

$$p_{k+1} = ReLU(\text{Random Walk Matrix} \times \text{Parameters} \times p_k)$$

L1-regularized PageRank

- The stationary vector of the non-linear PageRank diffusion corresponds to the optimal solution of the l1-regularized PageRank problem:

$$\text{minimize} \; \underbrace{\frac{1}{2}x^T Q x - \alpha x^T s}_{f(x)} + \underbrace{\rho\alpha\|Dx\|_1}_{g(x)}$$

where $Q = \alpha D + \frac{1-\alpha}{2} L$

Fountoulakis et al. Variational Perspective of Local Graph Clustering, Mathematical Programming, 2017

Properties of the l1-regularized optimal solution

- **Theorem**

- If the graph is unweighted then the number of nonzero nodes in the optimal solution is bounded by $1/\rho$.

- If the graph is weighted then the volume of nonzero nodes in the optimal solution is bounded by $1/\rho$.

Fountoulakis et al. Variational Perspective of Local Graph Clustering, Mathematical Programming, 2017

The solution path is monotonic

- **Theorem**

- Let $\hat{x}(\rho)$ be the solution of the l1-regularized problem as a function of ρ.

- Then $\hat{x}(\rho)$ is a component-wise monotone function

$$\hat{x}(\rho_0) \leq \hat{x}(\rho_1) \text{ for } \rho_0 > \rho_1$$

- The inequality becomes strict when a component is positive.

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Stage-wise for recovering the whole path

- Stage-wise algorithm

1) Choose i such that $\left|d_i^{-1}\nabla_i f(x_k)\right|$ is the largest among $[n]$

2) Update $[x_{k+1}]_i = [x_k]_i + \dfrac{\eta}{d_i}$

- Corollary

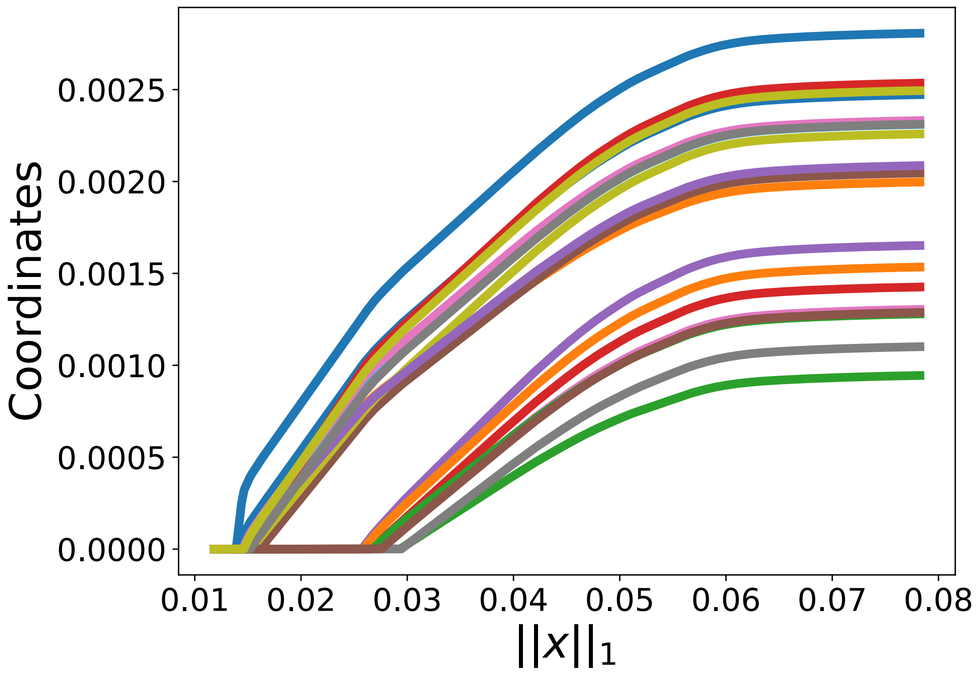
- The stage-wise algorithm converges to the l1-regularized solution path if we drag the step-size η of the algorithm to zero.

- The running time of stage-wise depends on the nonzero nodes and its neighbors and not on the size of the whole graph.

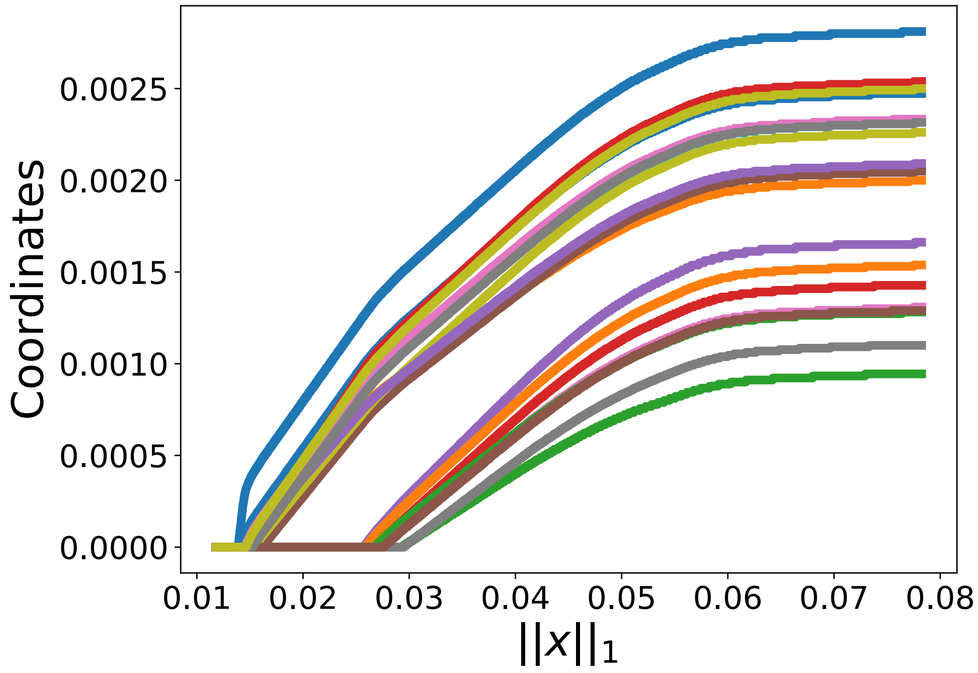
W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Stage-wise for recovering the whole path - example

L1-reg. Path



Stagewise path $\eta = 10^{-4}$



W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

What if we do not want to recover the whole path?

$$\text{minimize } \underbrace{\frac{1}{2}x^T Q x - \alpha x^T s}_{f(x)} + \underbrace{\rho\alpha\|Dx\|_1}_{g(x)}$$

Proximal gradient descent

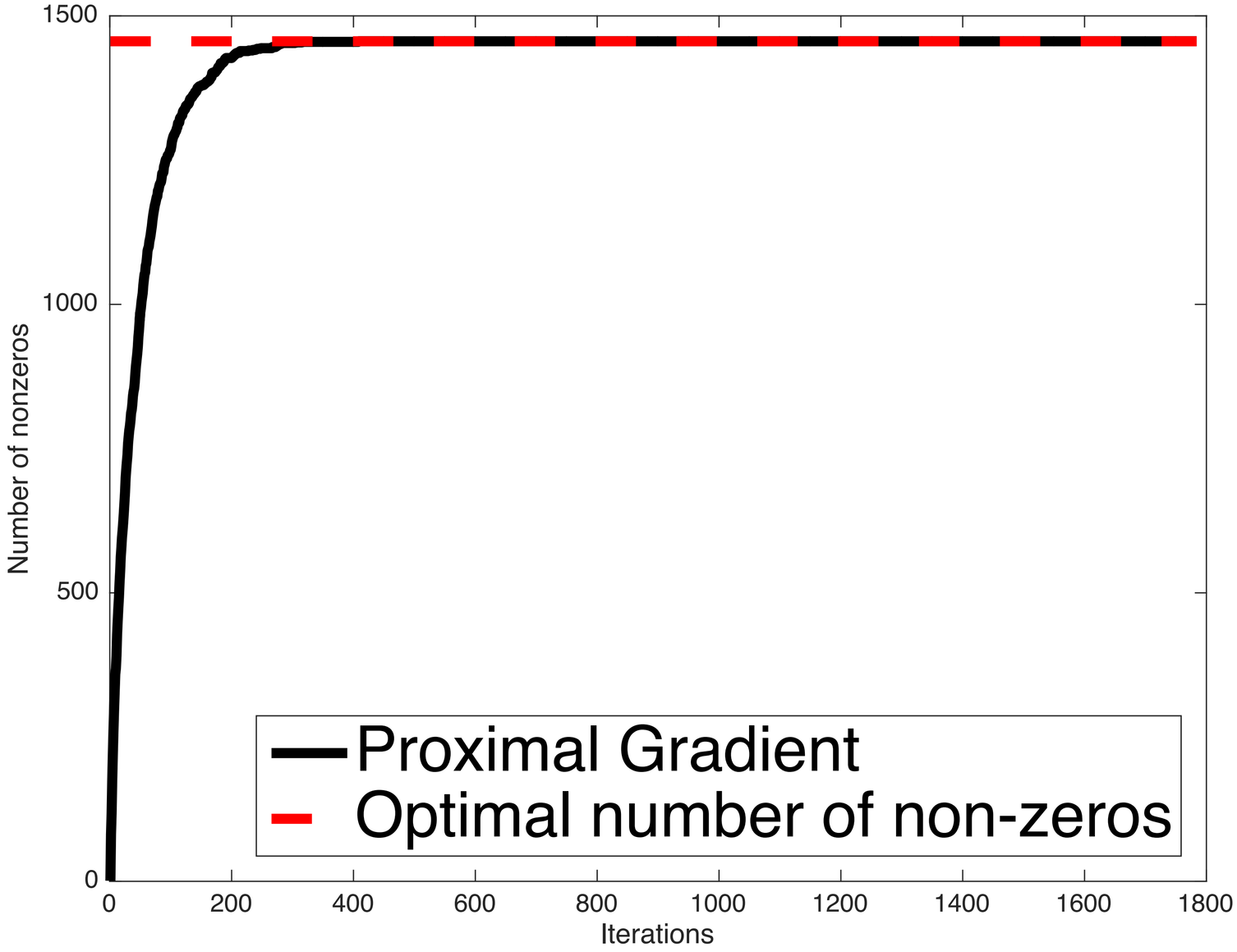
$$x_{k+1} := \operatorname{argmin}\ g(x) + \underbrace{f(x_k) + \langle \nabla f(x_k), x - x_k \rangle}_{\text{first-order Taylor approximation}} + \underbrace{\frac{1}{2}\|x - x_k\|_2^2}_{\substack{\text{upper bound on the} \\ \text{approximation error}}}$$

Requires careful implementation to avoid excessive running time

- Need to maintain a set of non-zero nodes
- Update x and gradient only for non-zero nodes and their neighbors at each iteration

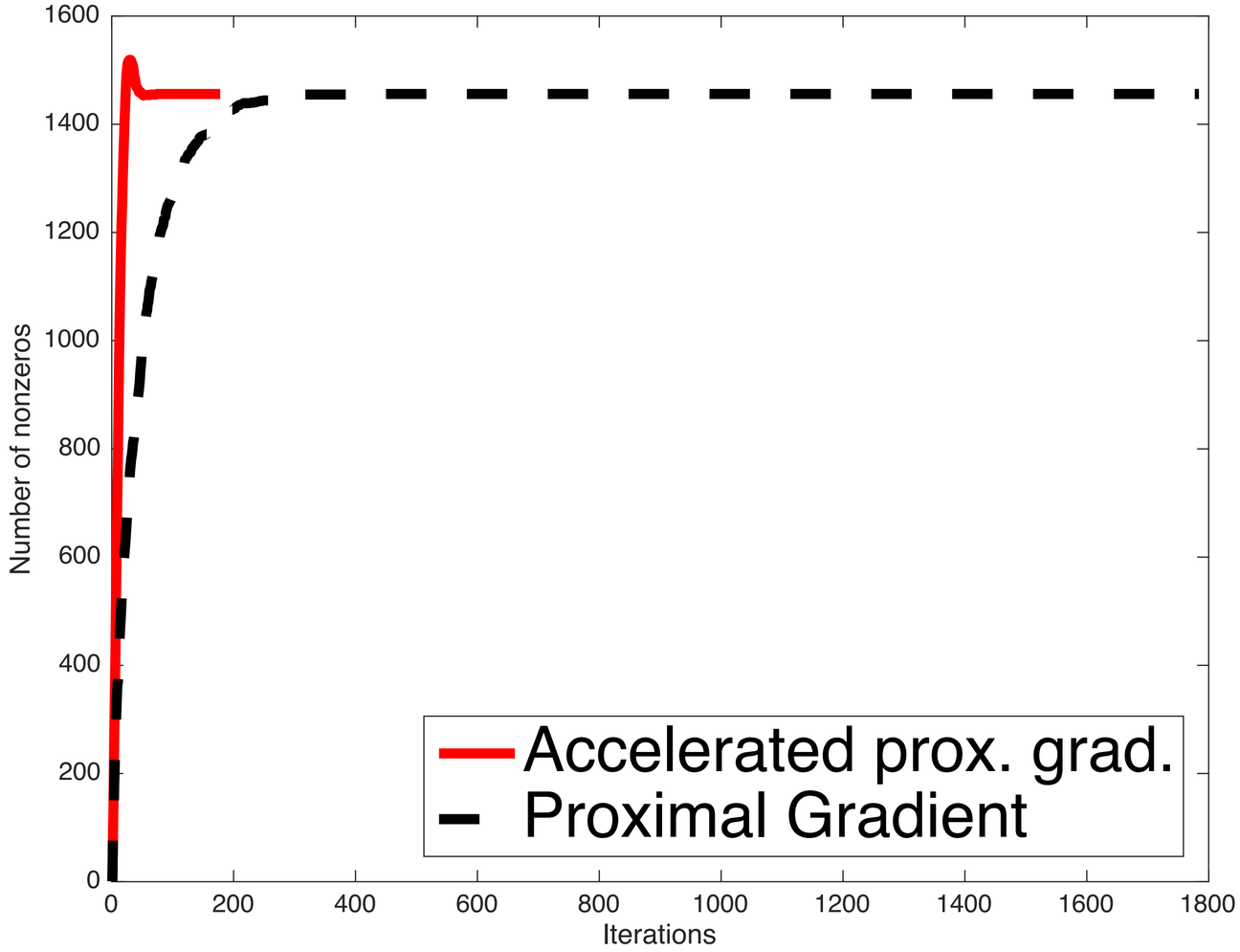
Fountoulakis et al. Variational Perspective of Local Graph Clustering, Mathematical Programming, 2017

Theorem: non-decreasing non-zero nodes



Fountoulakis et al. Variational Perspective of Local Graph Clustering, Mathematical Programming, 2017

Open problem: is accelerated prox. grad. a local algorithm?



Gradient descent running time

$$\tilde{\mathcal{O}}\left(\frac{\mathrm{vol}(\hat{S})}{\mu}\right)$$

Accel. gradient descent

$$\tilde{\mathcal{O}}\left(\frac{\mathrm{vol}(G)}{\sqrt{\mu}}\right)$$

$$\downarrow$$

$$\tilde{\mathcal{O}}\left(\frac{\mathrm{vol}(\hat{S})}{\sqrt{\mu}}\right)$$

- $\hat{S}$: support of optimal solution, i.e., non-zero nodes.
- μ strong convexity parameter of the problem.

Two ways to measure performance of the l1-regularized PageRank model

Average-case

- Performance under stochastic block model - recover a cluster using the output of l1-regularized PageRank.

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Worst-case

- Use conductance to measure quality of the output. Show that the output has conductance value similar to a target cluster around the seed node.

Fountoulakis et al. Variational Perspective of Local Graph Clustering, Mathematical Programming, 2017

Zhu et al. A local algorithm for finding well-connected clusters, ICML, 2013

Average-case guarantees

Average-case performance

Local random model

- Given a graph G with n nodes, let K be a target cluster inside G.
- Two nodes in K are connected with probability p
- Nodes in K are connected K^c with probability q.
- The rest of edges can be drawn using **any** other model.

Expected l1-regularized PageRank

- The optimal solution of the expected problem identifies the target cluster.

- **Theorem**

- Suppose that the seed node is selected from target cluster K. The optimal solution of

$$x^* := \text{argmin} \frac{1}{2} x^T \mathbb{E}[Q] x - \alpha x^T s + \rho \alpha \| \mathbb{E}[D] x \|_1$$

- satisfies

$$\text{supp}(x^*) = K$$

- as long as $\rho = \mathcal{O}\left(p/\bar{d}^2\right)$

- where $\bar{d}$ is the expected degree of nodes in the target cluster.

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Results for l1-regularized PageRank for noisy data

- In practice, we do not have access to the expected graph. We are given a realization of the local random model that includes “noise”, i.e., edges from the target cluster to the rest of the graph.

- We have two results for the noisy case.

- **First result.**

- Zero false negatives.

- Bounded false positives.

- **Second result.**

- With additional assumptions on the seed nodes we can show exact recovery.

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

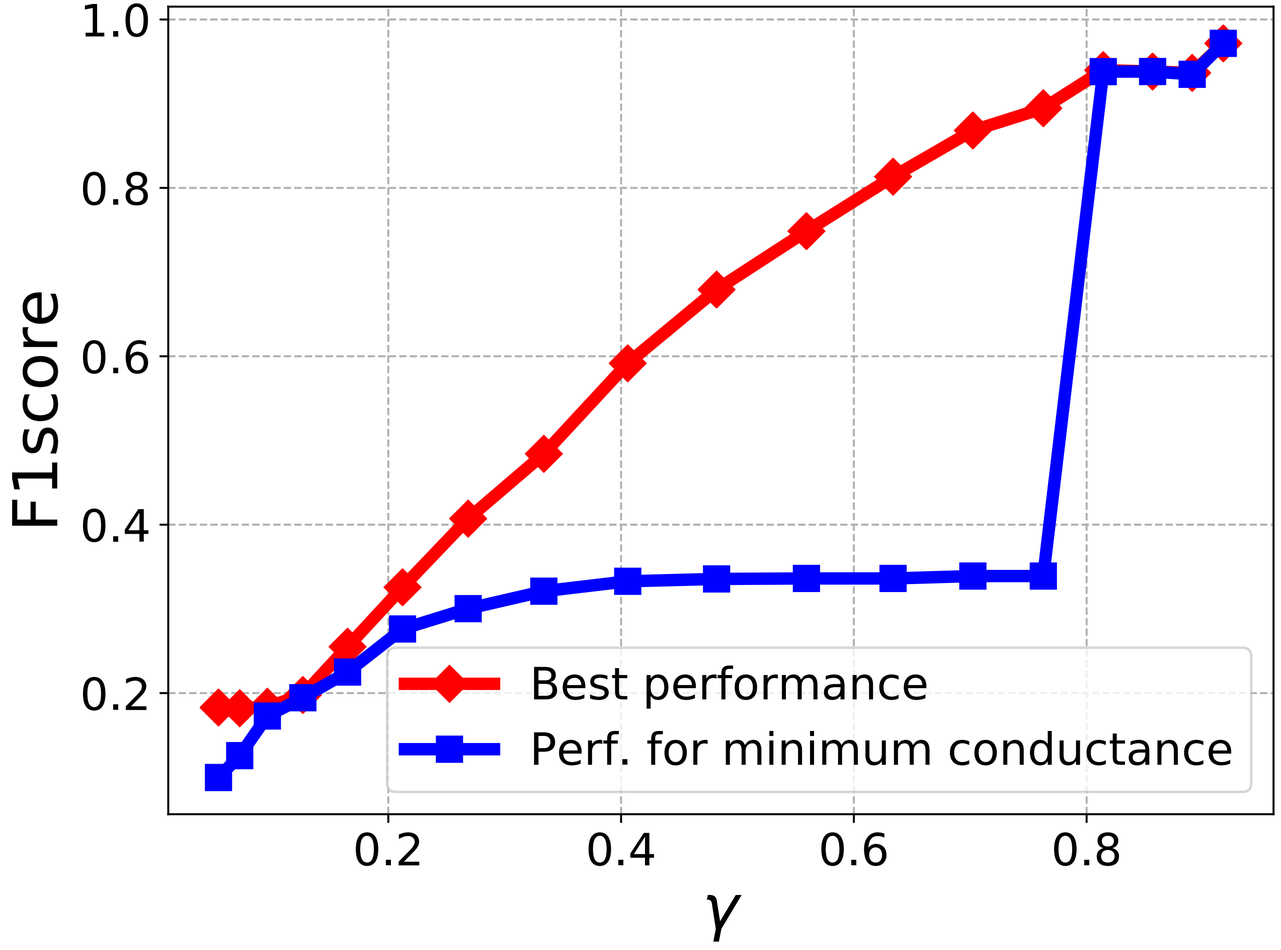
Results for l1-regularized PageRank for noisy data

- **Theorem (bounded false positives)**
- Suppose $p^2 k \geq \mathcal{O}(\log k)$, where k is the size of the target cluster.
- and $\rho = \mathcal{O}\left(\gamma p / \bar{d}^2\right)$
- where $\gamma = pk/\bar{d}$, i.e., the probability of staying inside the target cluster in one step.
- Then with probability $1 - 6\exp(-\mathcal{O}(p^2 k))$ the optimal solution of the realized problem has **zero false negatives** and the false positives are bounded

$$\text{vol}(FP) \leq \text{vol}(K)\left(\mathcal{O}\left(\frac{1}{\gamma^2}\right) - 1 \right)$$

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Results for l1-regularized PageRank for noisy data



Definitions

$$\Phi(B) := \left(\frac{\text{number of edges leaving } B}{\text{sum of edges of vertices in } B} \right)$$

Assuming B is the smaller part of the graph

where $\gamma = \dfrac{pk}{\bar{d}}$, i.e., the probability of staying inside the target cluster in one step.

Results for l1-regularized PageRank for noisy data

- **Theorem (exact recovery)**

- Let $q = \mathcal{O}(1/n)$

- Then with probability at least $1 - \mathcal{O}(e^{-k})$ there exists a good seed node such that if we use that seed node we get

$$\text{supp}(\hat{x}) = K$$

- As long as

$$d_j \geq \mathcal{O}\left(\frac{1}{\gamma p}\right) \ \forall j \in K^c$$

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Results for l1-regularized PageRank for noisy data

- The assumption that $q = \mathcal{O}(1/n)$

- implies that there are constant number of edges leaving the cluster, which sounds artificial.

- but it is not, because it also covers the case were the size of the target cluster is $k = \mathcal{O}(1)$

- This is a realistic local graph clustering setting where we attempt to recover a very small target cluster of constant size with constant number of edges leaving the cluster.

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Worst-case guarantees

Some definitions

- **Conductance of target cluster B:**

$$\Phi(B) := \left(\frac{\text{number of edges leaving } B}{\text{sum of edges of vertices in } B} \right)$$

Assuming B is the smaller part of the graph

- **Internal connectivity of target cluster B**

$IC(B) :=$ the minimum conductance of the subgraph induced by B

Worst-case performance

- **Theorem (by Zhu et al.)**
- Assume that the internal connectivity of the target cluster K is larger than its conductance

$$\frac{IC^2(K)}{\Phi(K)\log \mathrm{vol}(K)} \geq \Omega(1)$$

- False positives are bounded by

$$\mathrm{vol}(FP) \leq \mathcal{O}\left(\frac{\Phi(K)}{IC(K)}\right)\mathrm{vol}(K)$$

- True positives are bounded by

$$\mathrm{vol}(FN) \leq \mathcal{O}\left(\frac{\Phi(K)}{IC(K)}\right)\mathrm{vol}(K)$$

Zhu et al. A local algorithm for finding well-connected clusters, ICML, 2013

Compare average- and worst-case

	False Positives	**False Negatives**
Average-case	$\text{vol}(FP) \leq \text{vol}(K)\left(\mathcal{O}\left(\frac{1}{\gamma^2}\right) - 1\right)$	zero
Worst-case	$\text{vol}(FP) \leq \text{vol}(K)\mathcal{O}((1-\gamma)\log k)$	$\text{vol}(FN) \leq \text{vol}(K)\mathcal{O}((1-\gamma)\log k)$

- The average-case result on FP is stronger for large values of γ.
- Also for the average-case we can also prove exact recovery.

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Comparison to planted cluster model

- Example, $p = 1$ and $q = \mathcal{O}(\log n/n)$

- Using semidefinite programming one can achieve exact recovery as long as $k \geq \mathcal{O}(\log n)$,

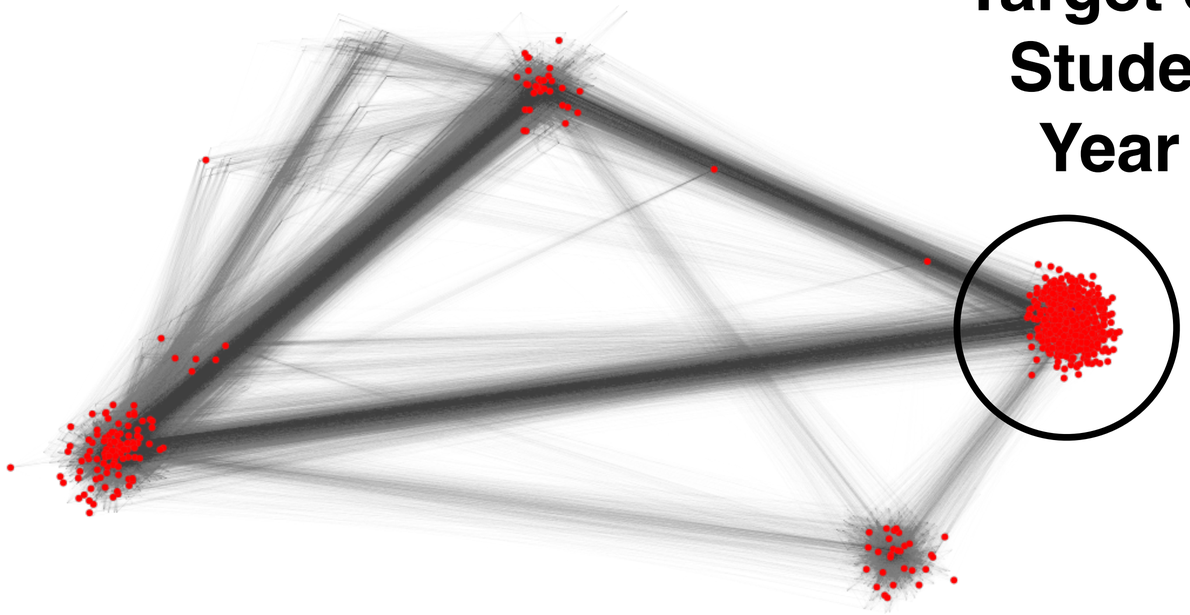
- while our results guarantee zero false negative and a constant proportion of false positives.

- However, our model is not allowed to touch the whole graph.

W. Ha, K. Fountoulakis, M. Mahoney. Statistical Guarantees of Local Graph Clustering. AISTATS-2020

Combinatorial Diffusion: Capacity Releasing Diffusion

Problem: spectral diffusions might leak mass



Target cluster:
Students of
Year 2008

Red nodes: output of the algorithm

ℓ_1-regularized PageRank (best tuning)
Precision=0.73, Recall=0.91

Data: Facebook Colgate University, A. L. Traud, P. J. Mucha and M. A. Porter, Physica A, 391(16), 2012

Solving the problem of spreading mass indiscriminately by gradual release of edge capacity

Spectral diffusions

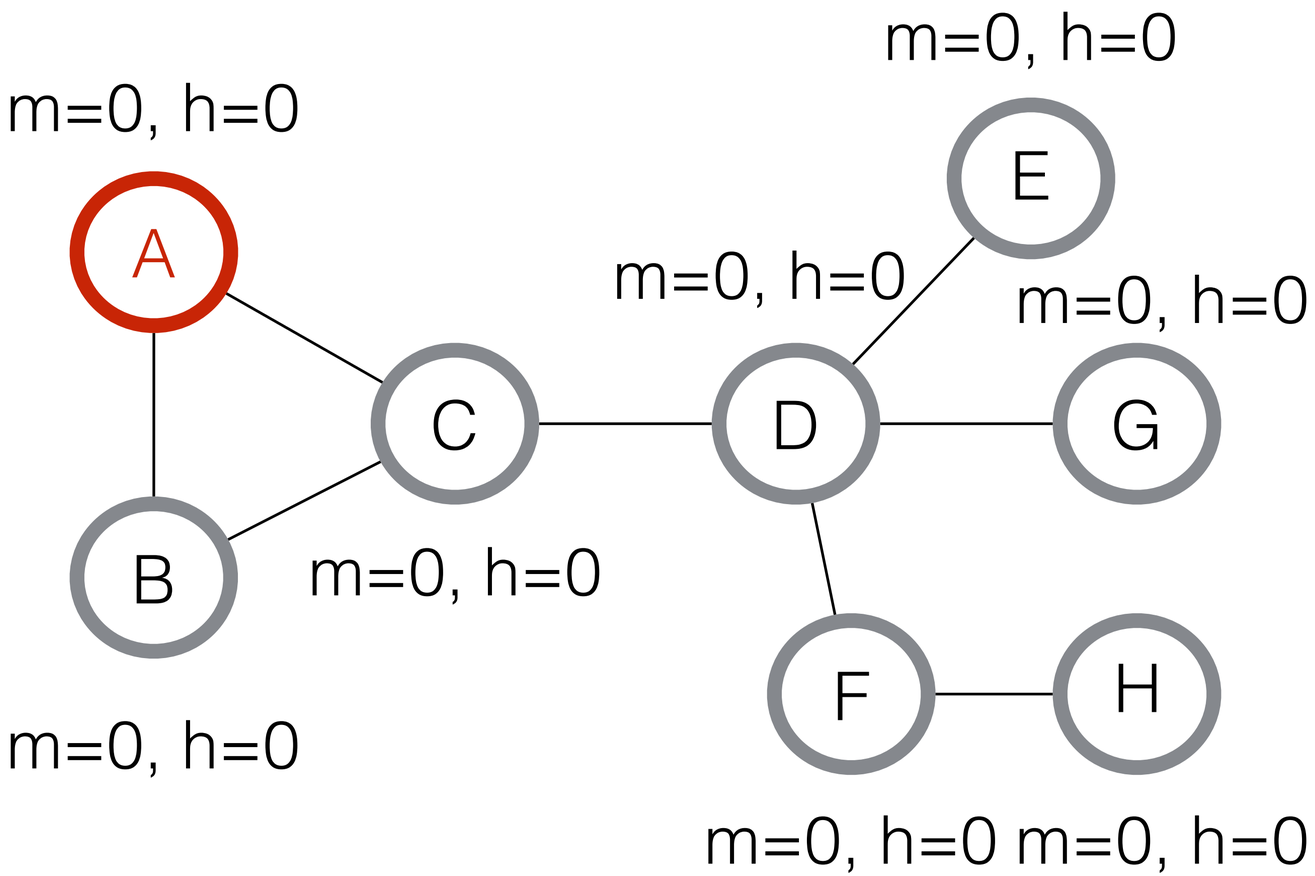
- Even distribution of the residual probability mass to neighbors

Capacity Releasing Diffusion

- Controls the amount of mass to be send over an edge by using the height “h” of a node
- In theory this results in bounded mass leaked outside of the target cluster
- In practice this results in much better precision and recall

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



Maintain mass “m” and height “h” for each node

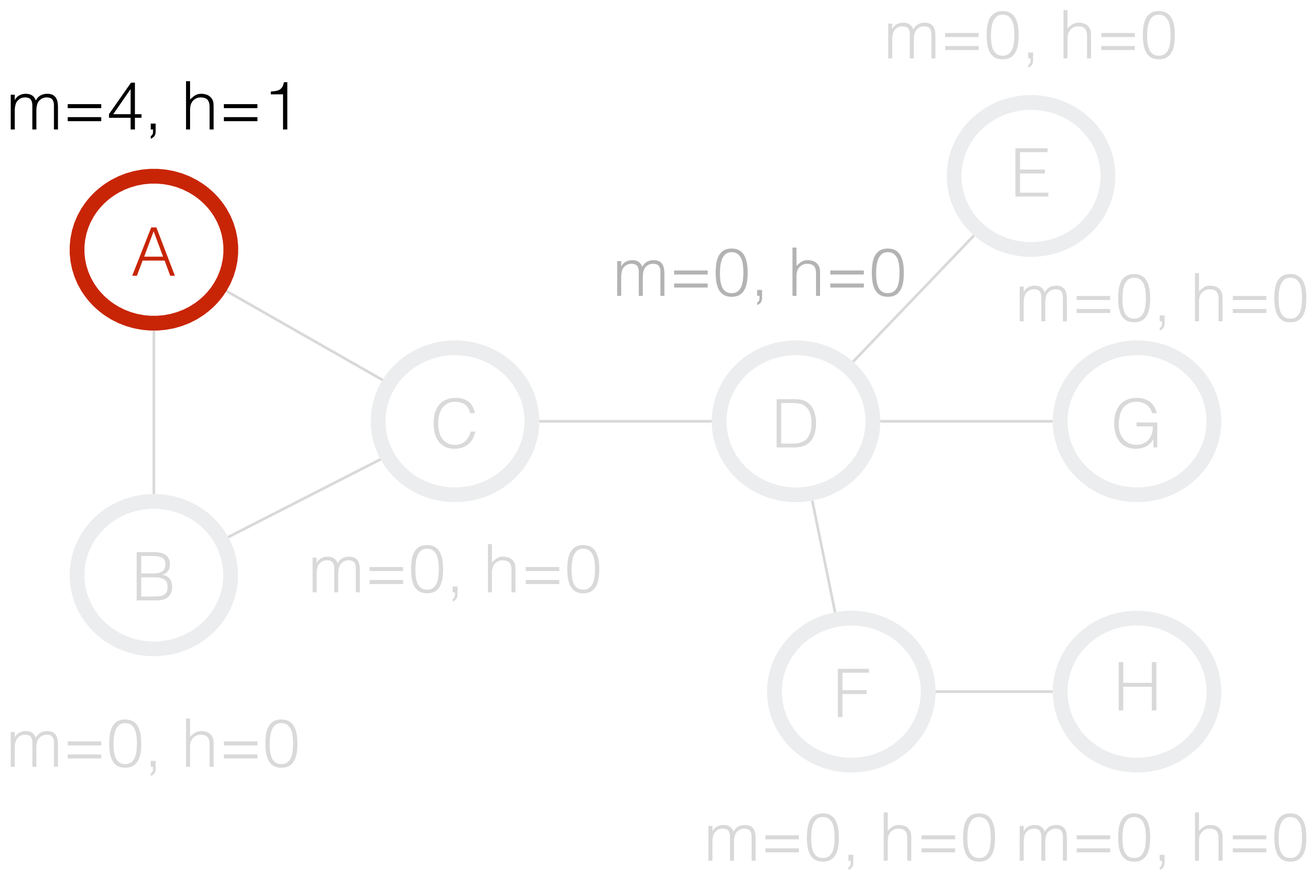
degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

m(v) <= 2deg(v) for all nodes v

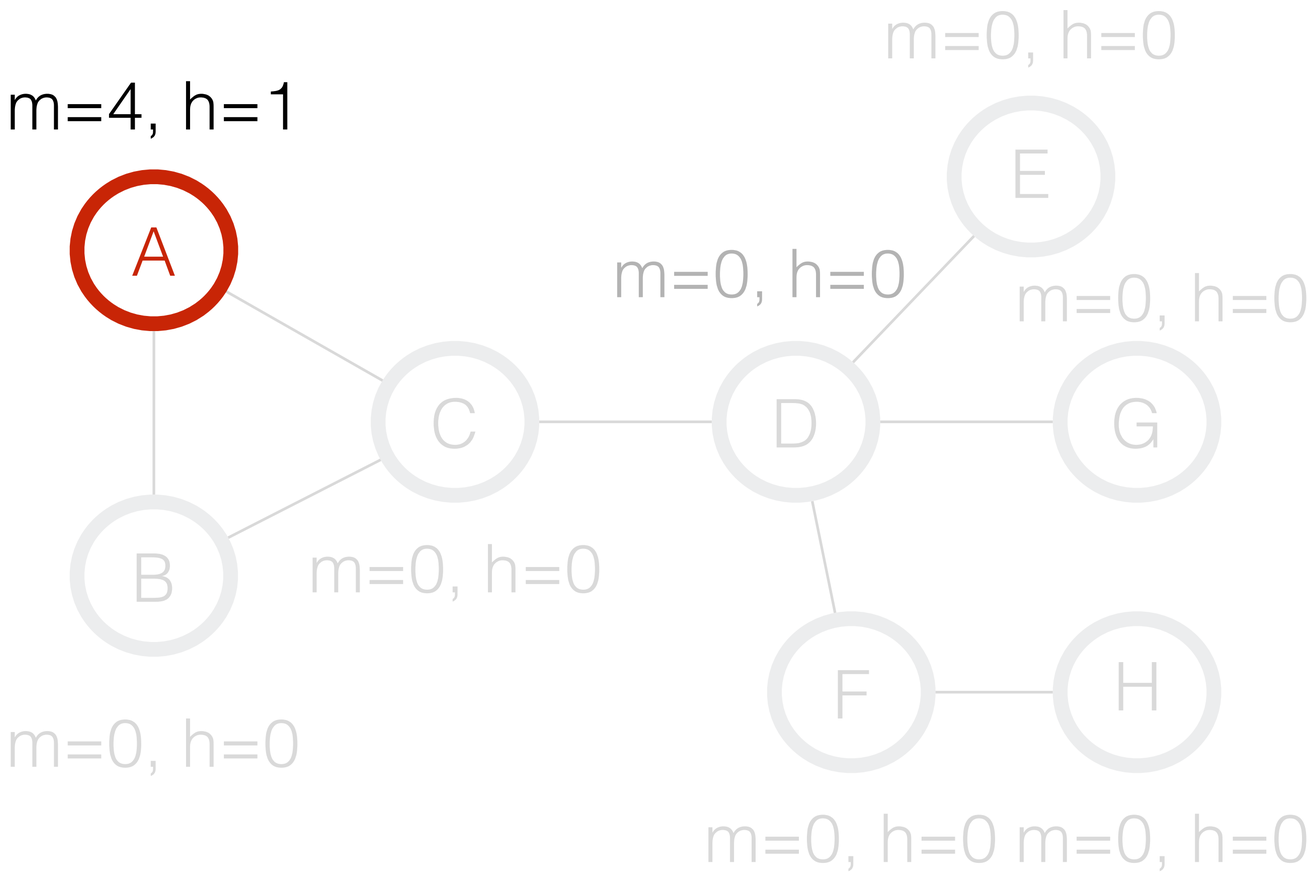
Push excess mass to unsaturated nodes with lower height

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

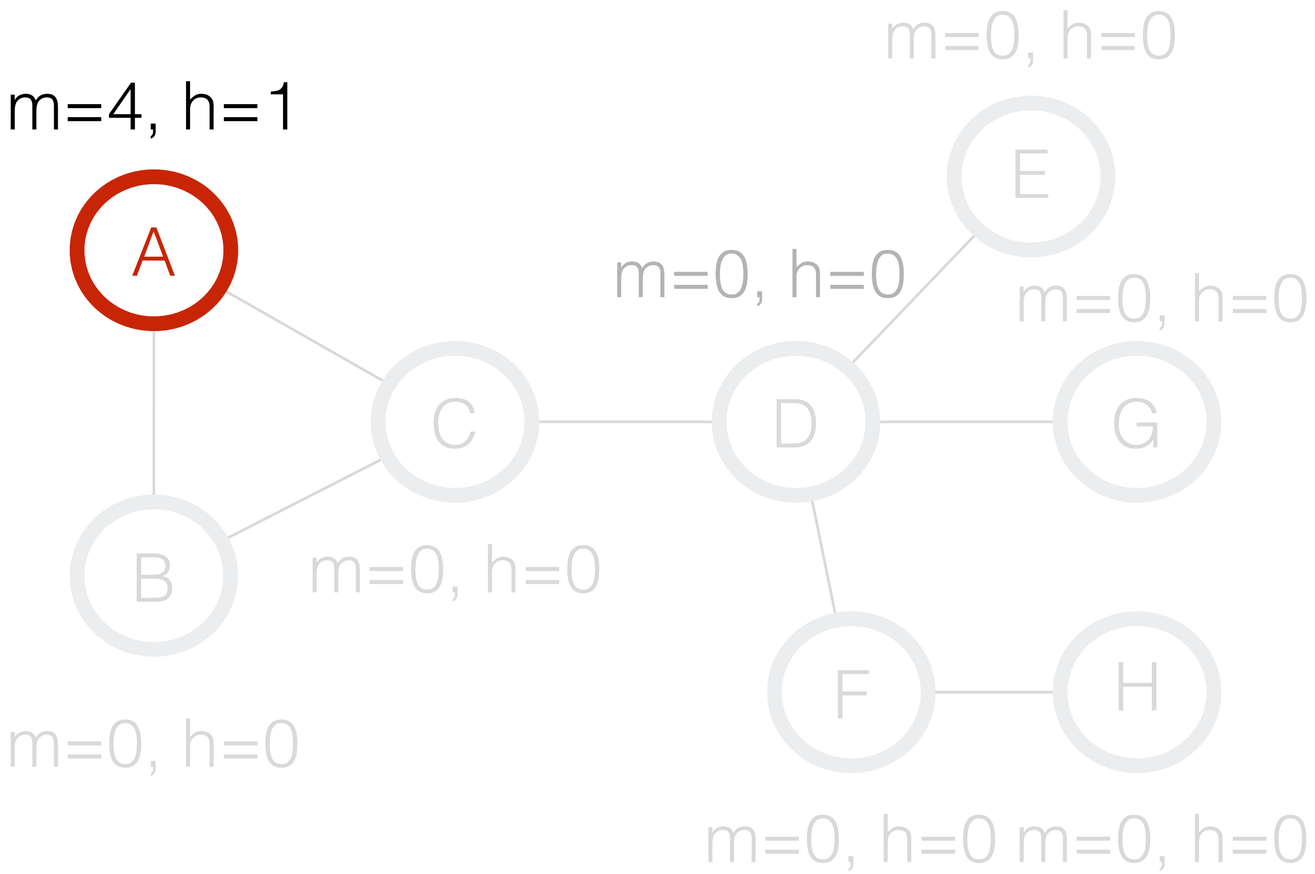
Push excess mass to unsaturated nodes with lower height

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

Pick node A (has excess mass)

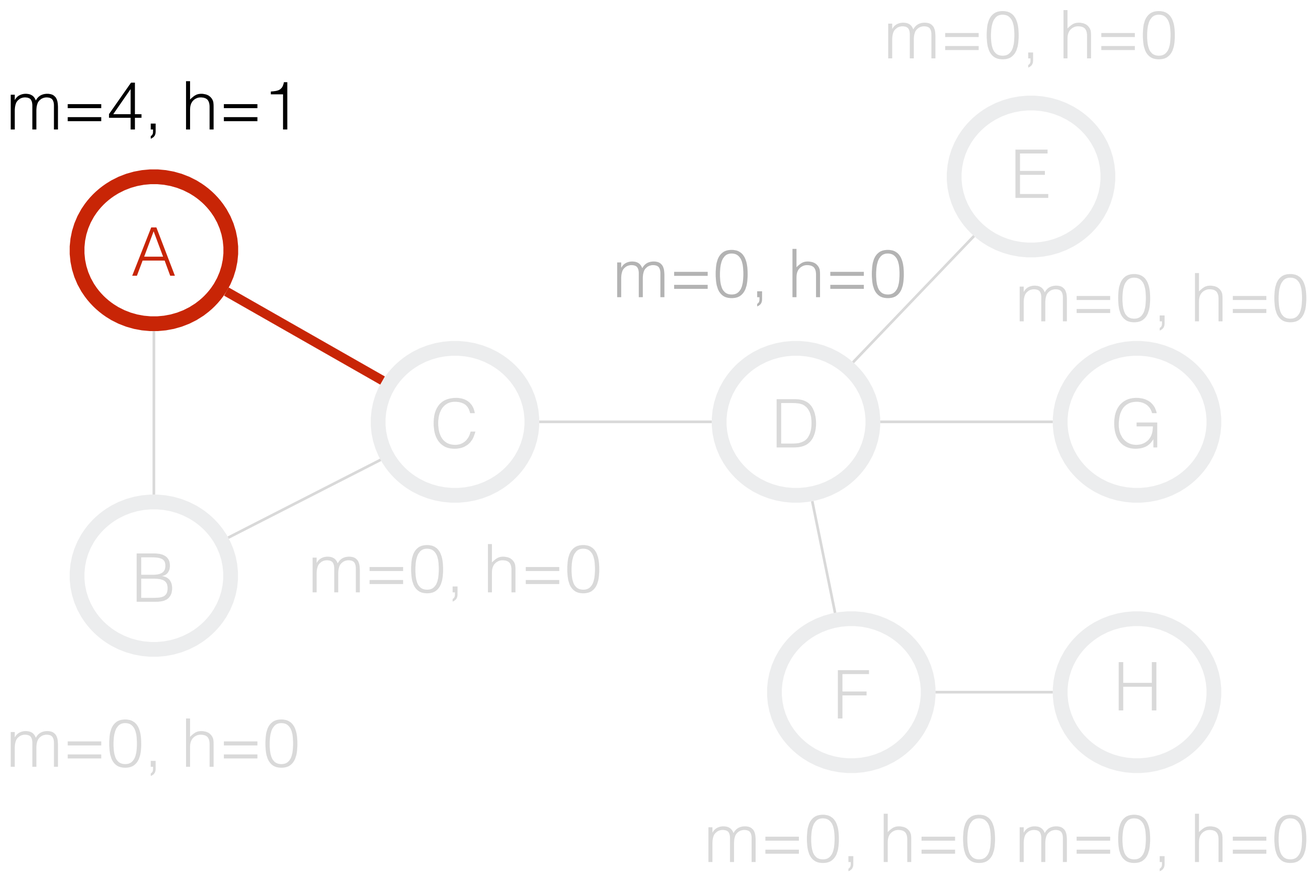
and a neighbor of A with lower height “h”

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

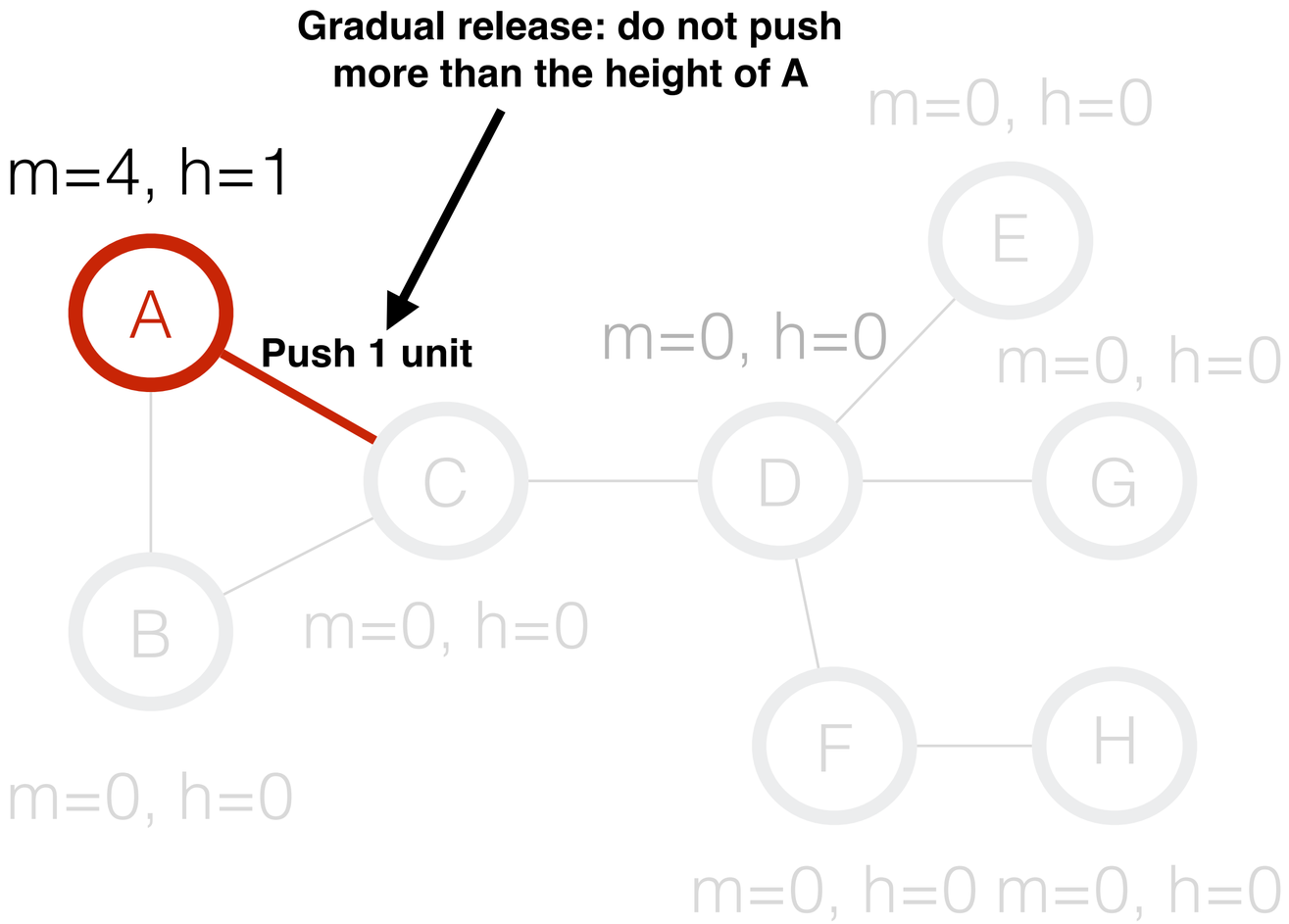
Pick node C

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

Pick node A (has excess mass)

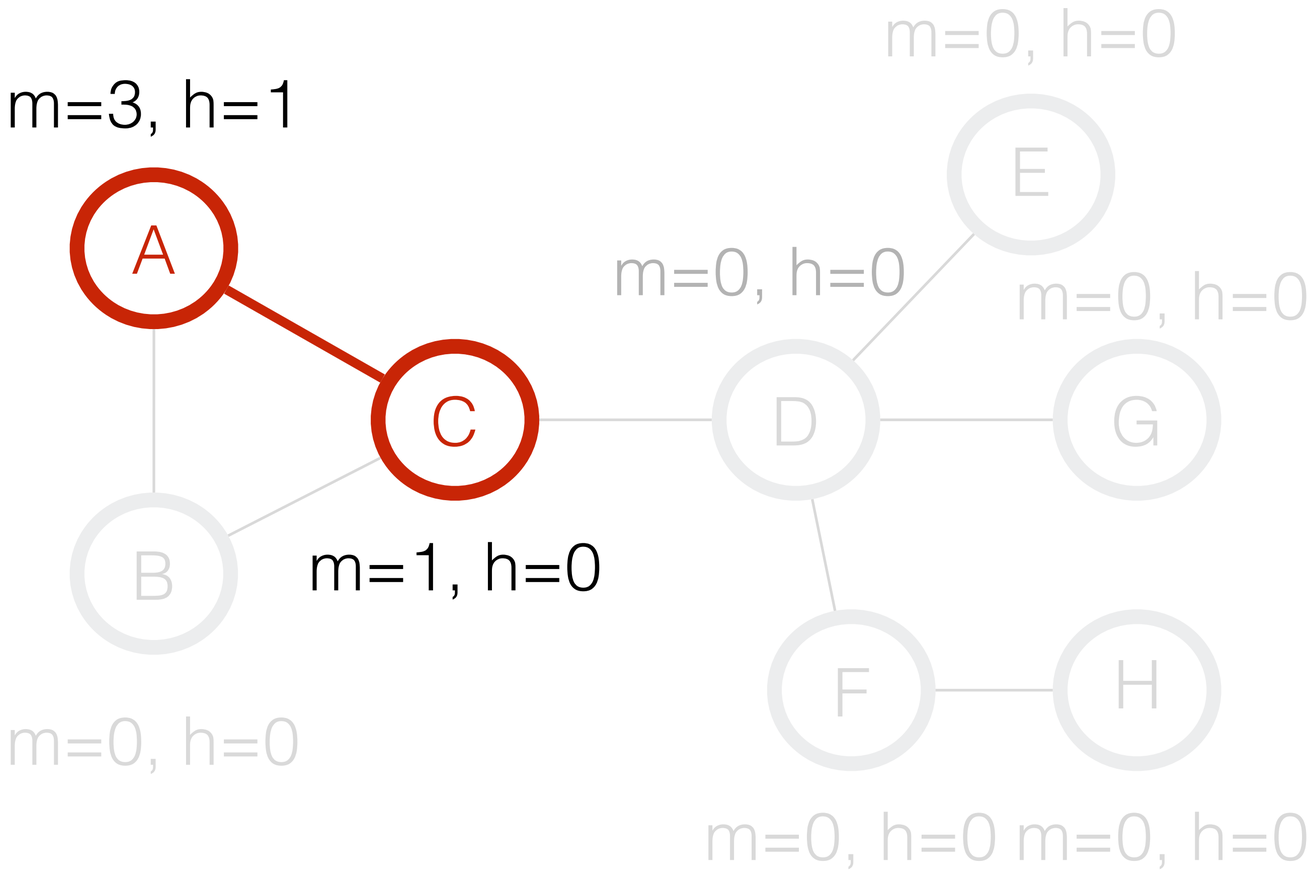
Push at most "h" flow to a chosen neighbor

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

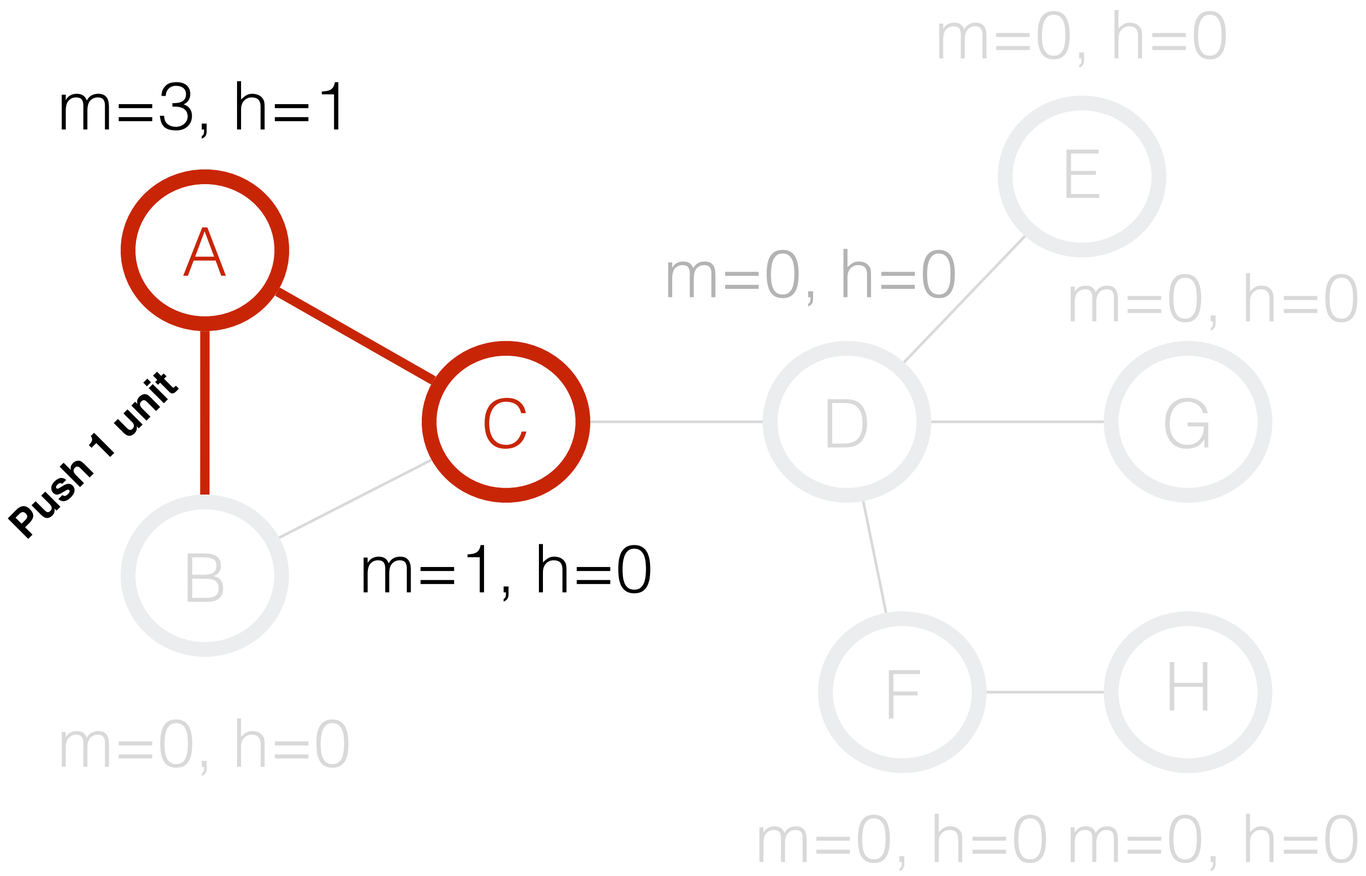
Push excess mass to unsaturated nodes with lower height

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

Pick node A (has excess mass)

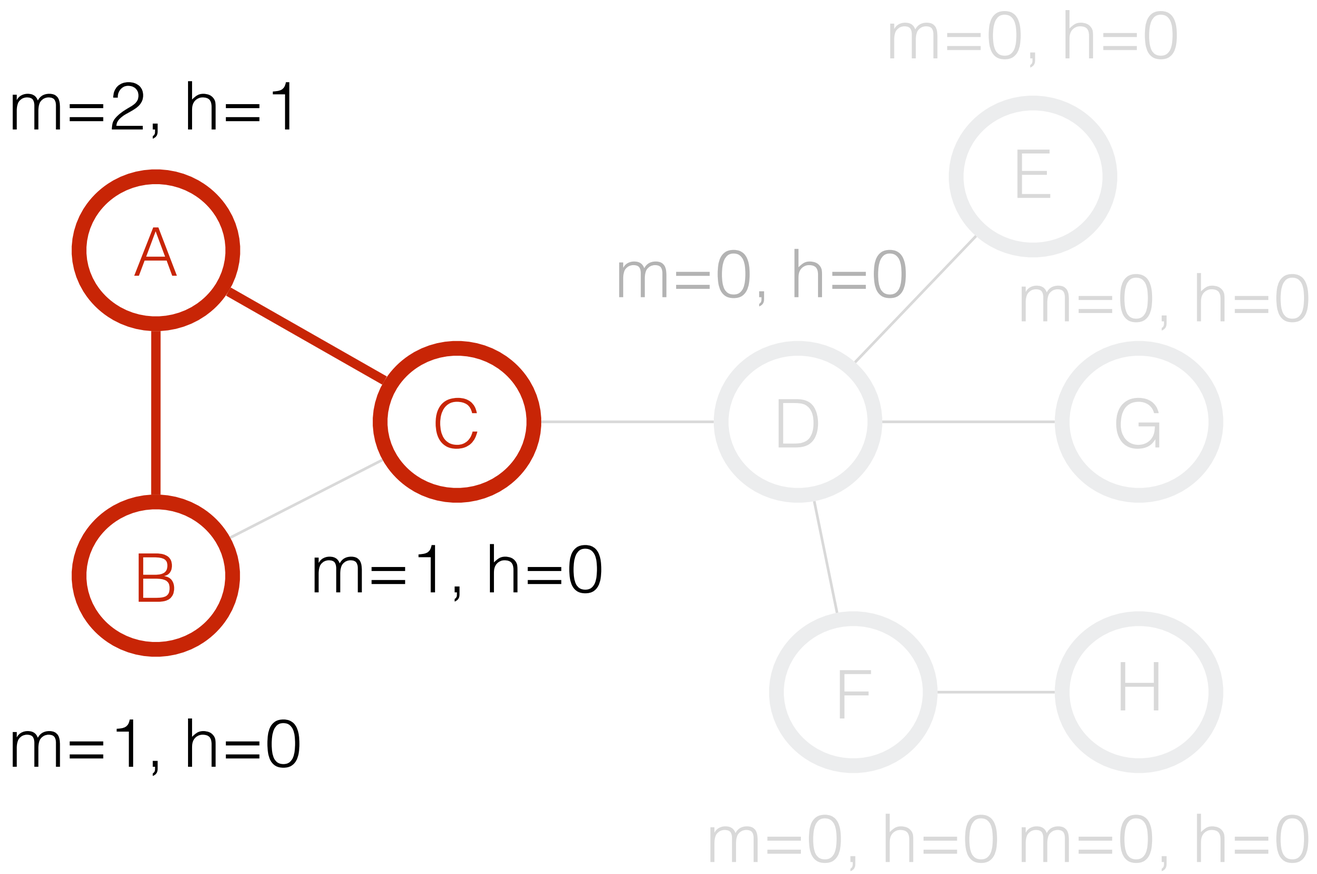
and a new edge of node A of residual flow less than “h”

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

m(v) <= 2deg(v) for all nodes v

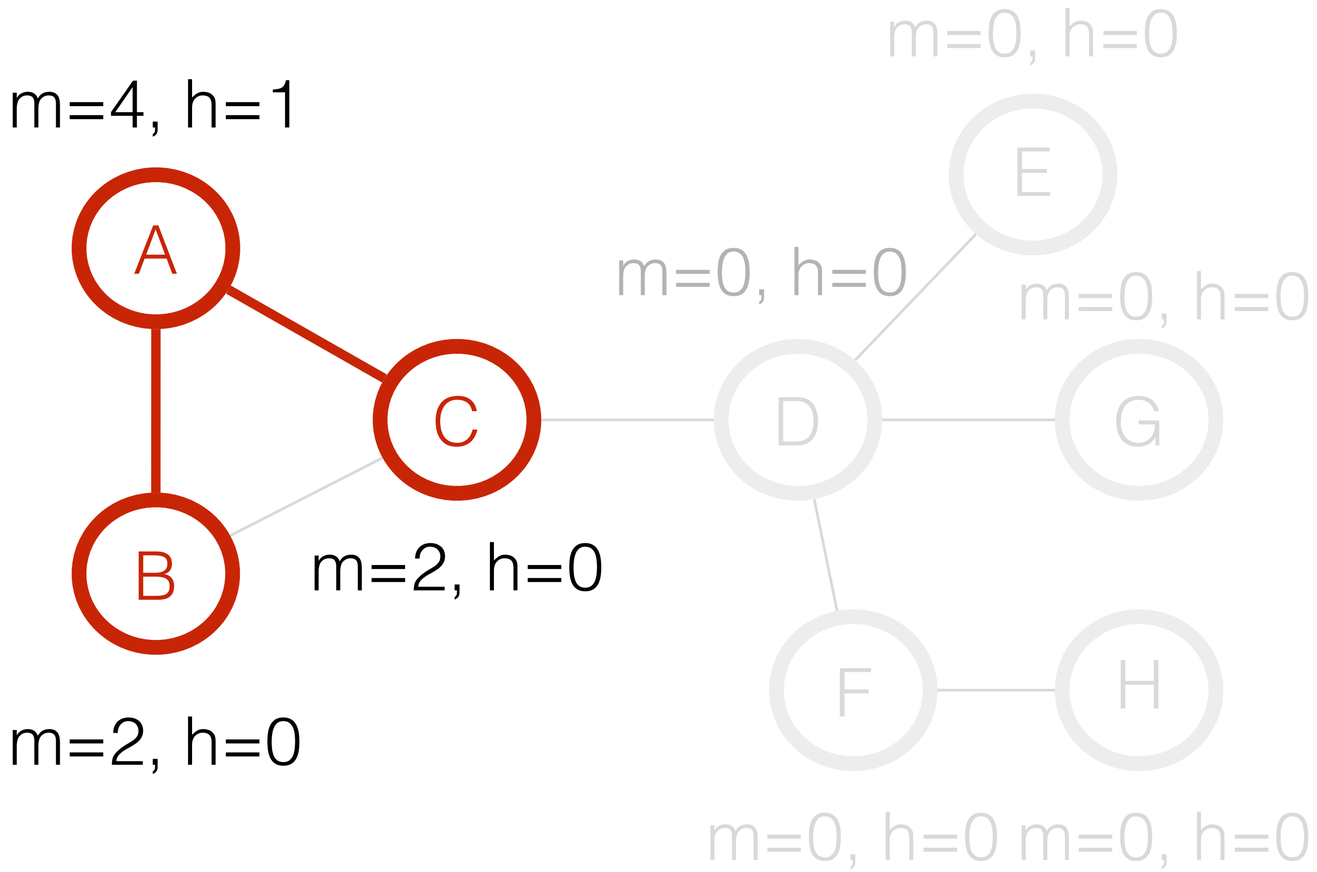
Push excess mass to unsaturated nodes with lower height

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

m(v) <= 2deg(v) for all nodes v

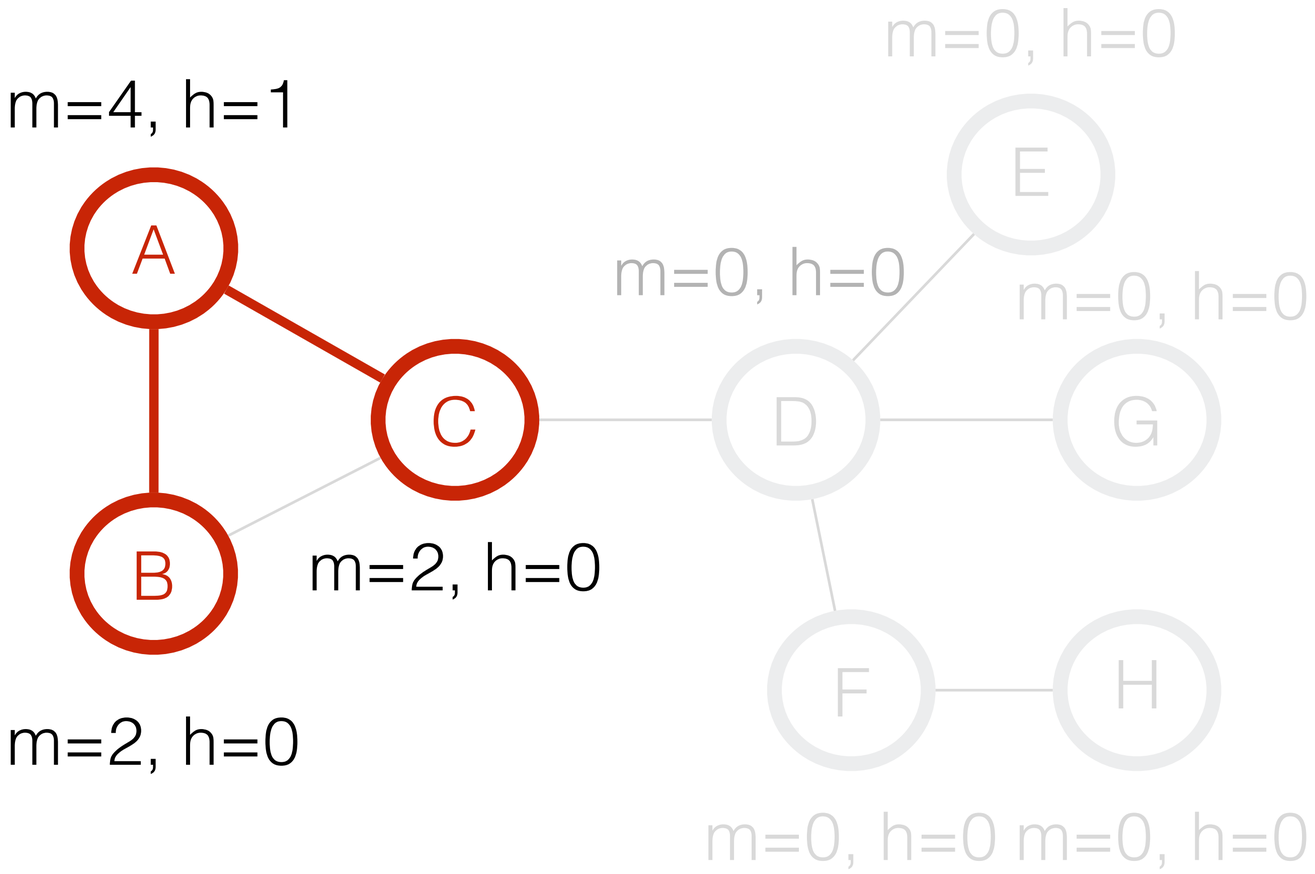
Push excess mass to unsaturated nodes with lower height

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

Pick node A (has excess mass)

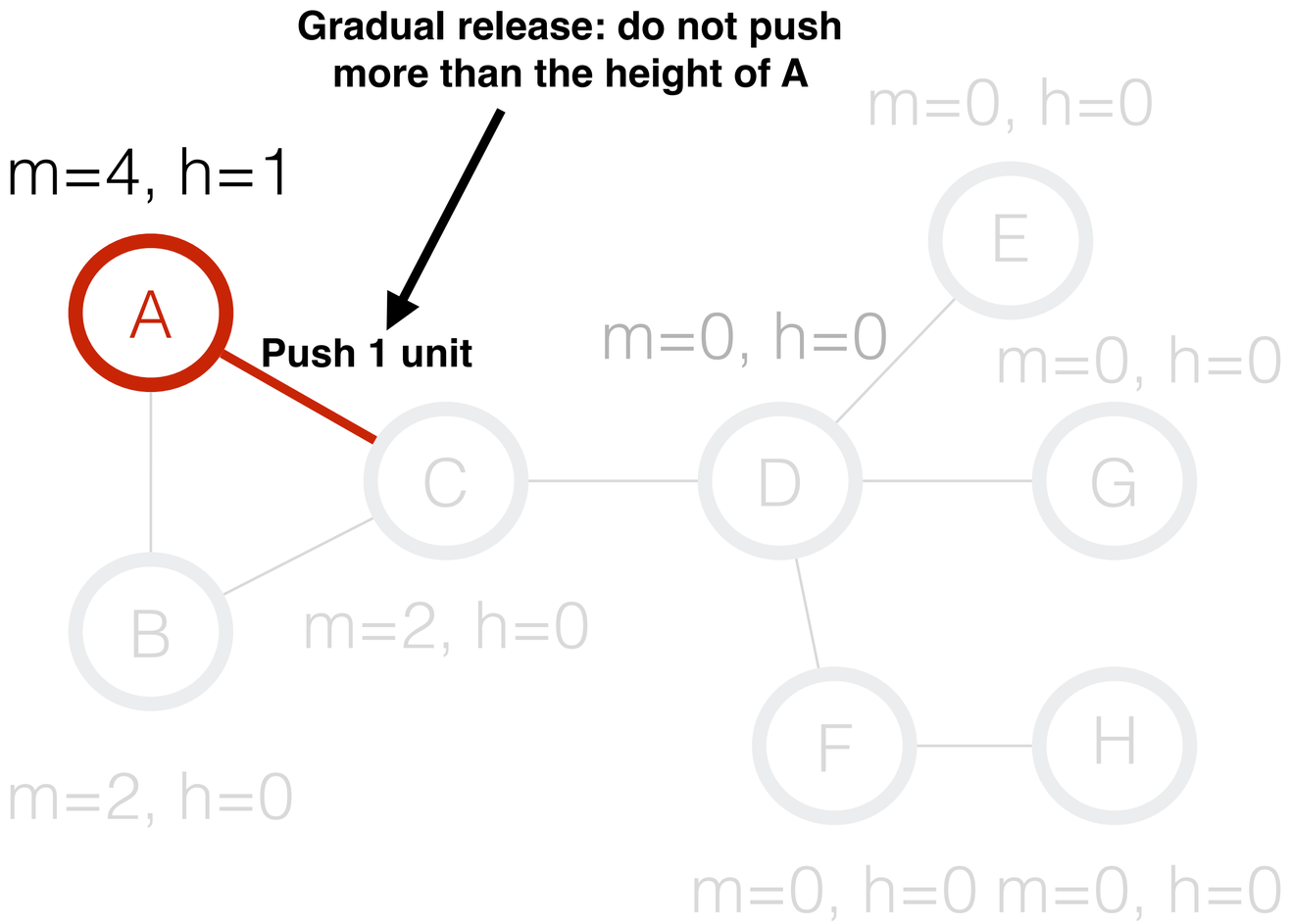
and a neighbor of A with lower height “h”

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

Pick node A (has excess mass)

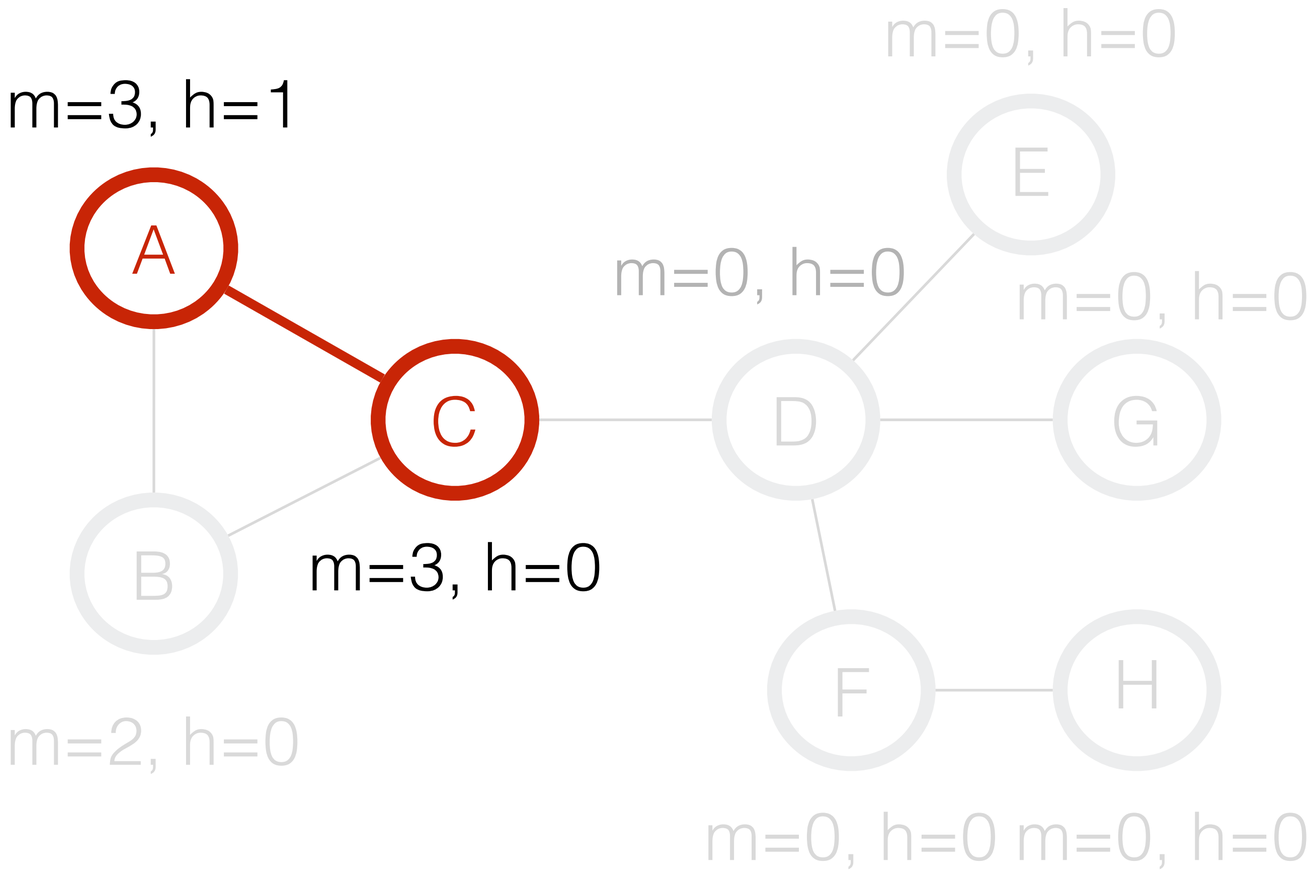
Push at most "h" flow to a chosen neighbor

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

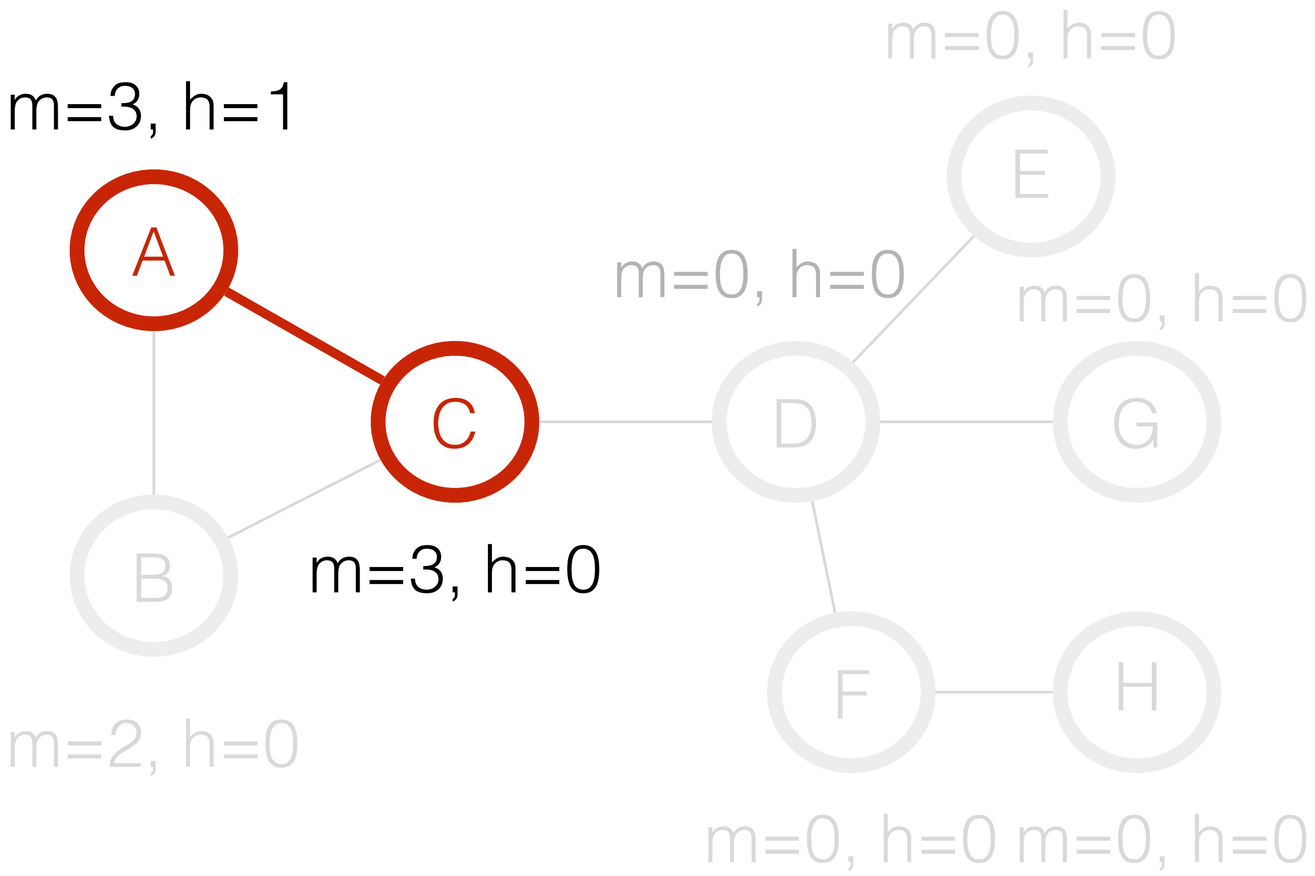
Note C has excess it has to be added to the candidate nodes

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

Pick node C (has excess mass)

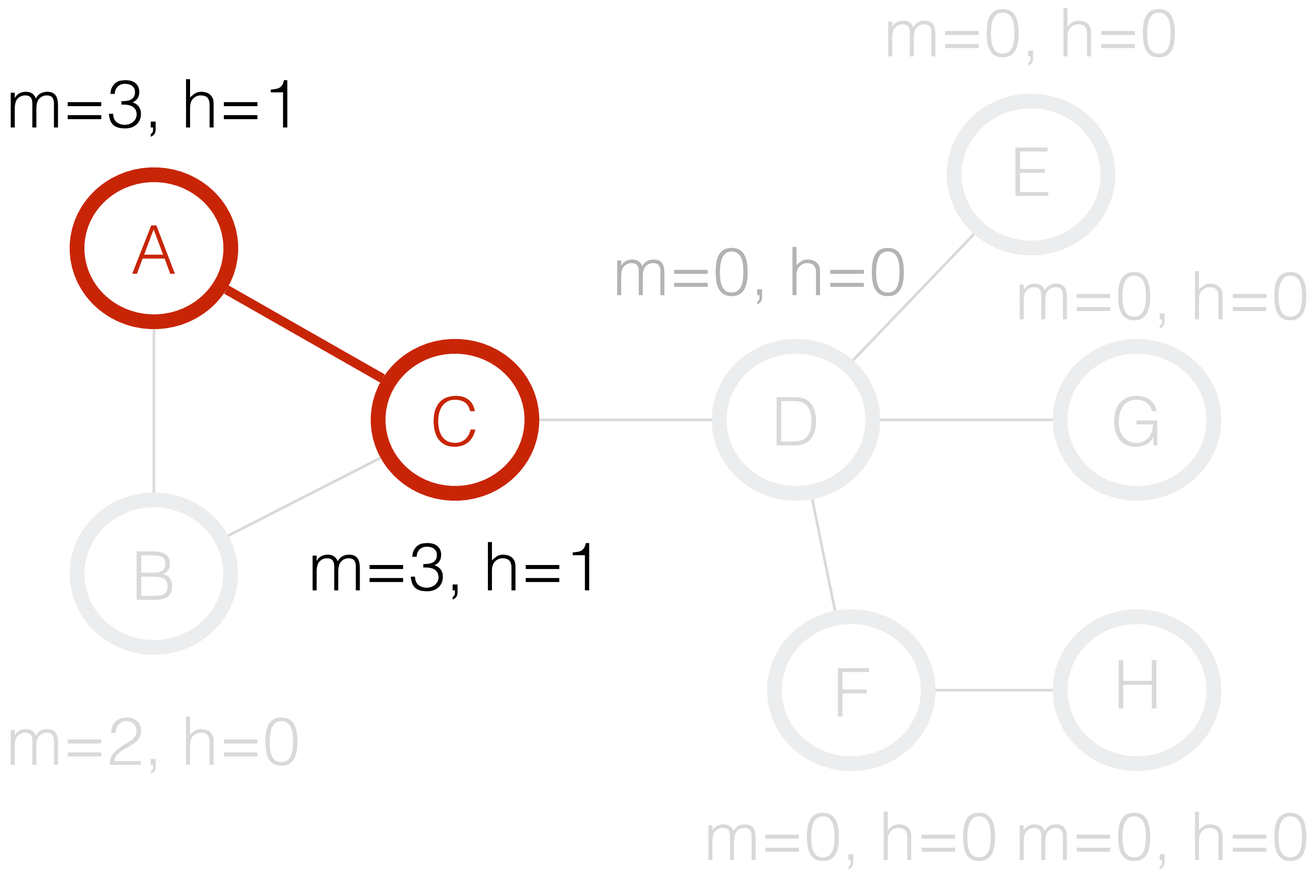
and a neighbor of C with lower height “h”

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

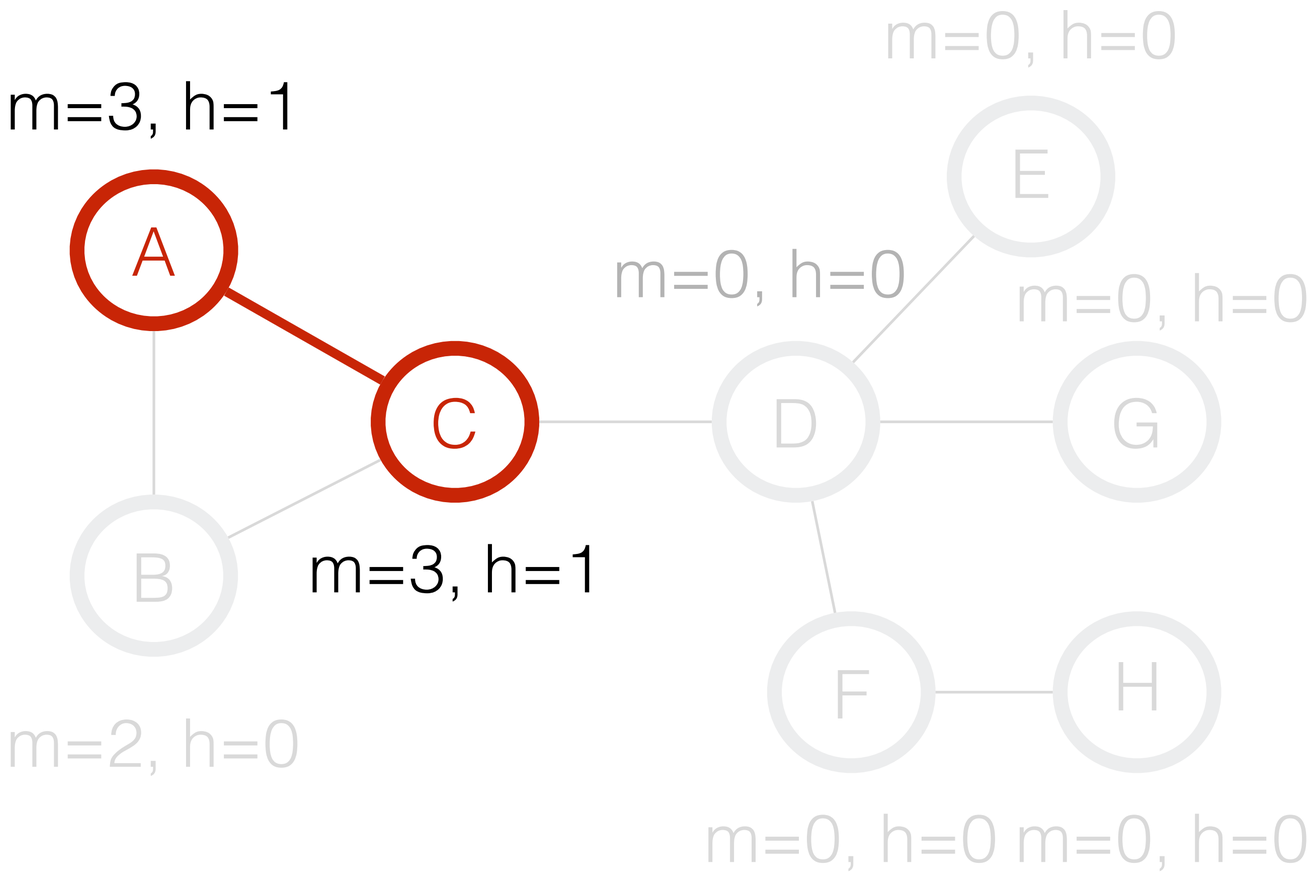
There is no neighbor of C with lower height so increase the height of C by 1

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Capacity Releasing Diffusion algorithm



degree(v): #edges of node v

Saturated nodes: m(v) >= deg(v)

Excess mass = max(m(v) - deg(v),0)

Algorithm

Overflow the seed: m(A) = 2deg(A)

Iterate

Repeat until there is no node with excess mass

m(v) <= deg(v) for all nodes v

Overflow: m(v) = 2m(v)

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Theoretical comparison to spectral diffusions

- **Conductance of target B ("noise"):** $\Phi(B) := \left(\frac{\text{number of edges leaving } B}{\text{sum of edges of vertices in } B} \right)$ Assuming B is the smaller part of the graph

- **Internal connectivity ("signal") of target B**

$$IC(B) := \text{the minimum conductance of the subgraph induced by } B$$

Weaker assumptions

- Theoretical bound on FP/FN needs: "signal" polylog stronger than "noise", as opposed to: quadratically stronger for spectral methods

Better worst-case guarantees

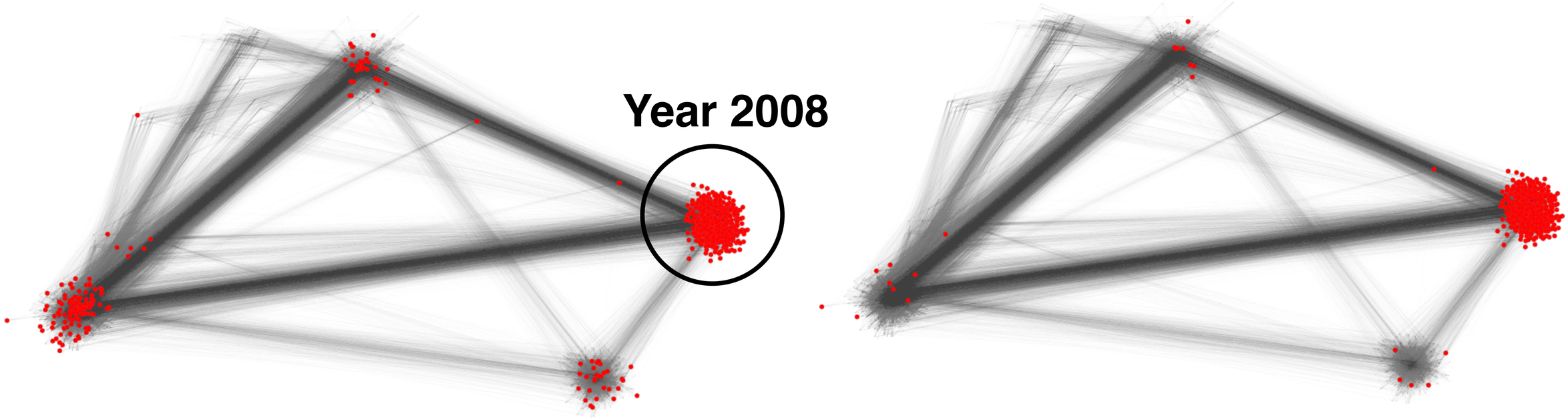
- Output A satisfies $\Phi(A) \leq \mathcal{O}(\Phi(B))$, as opposed to $\Phi(A) \leq \mathcal{O}(\Phi(B)/IC(B))$

Better running time

- The running time is $1/IC(B)$ times faster than spectral

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Example on Facebook Colgate University social network

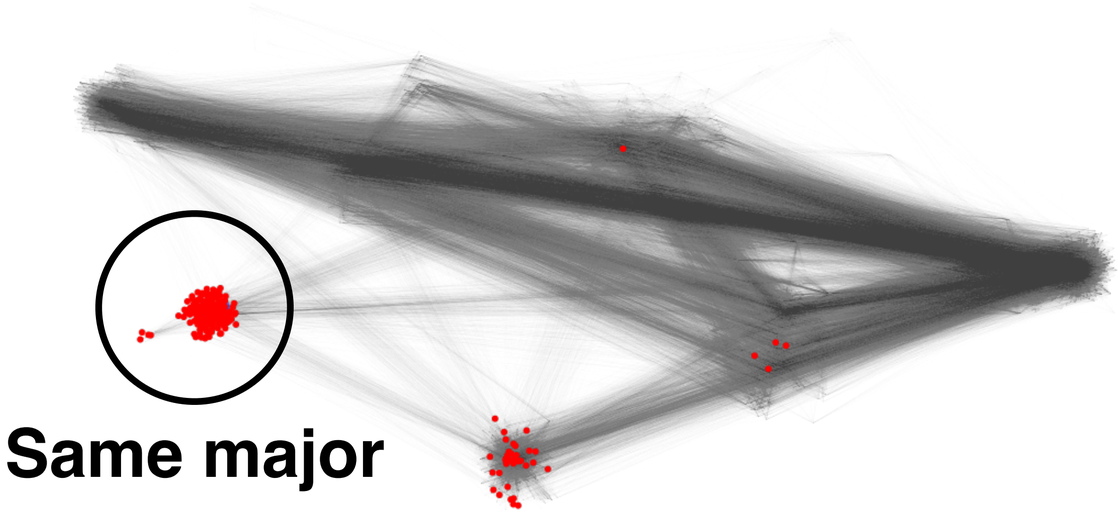


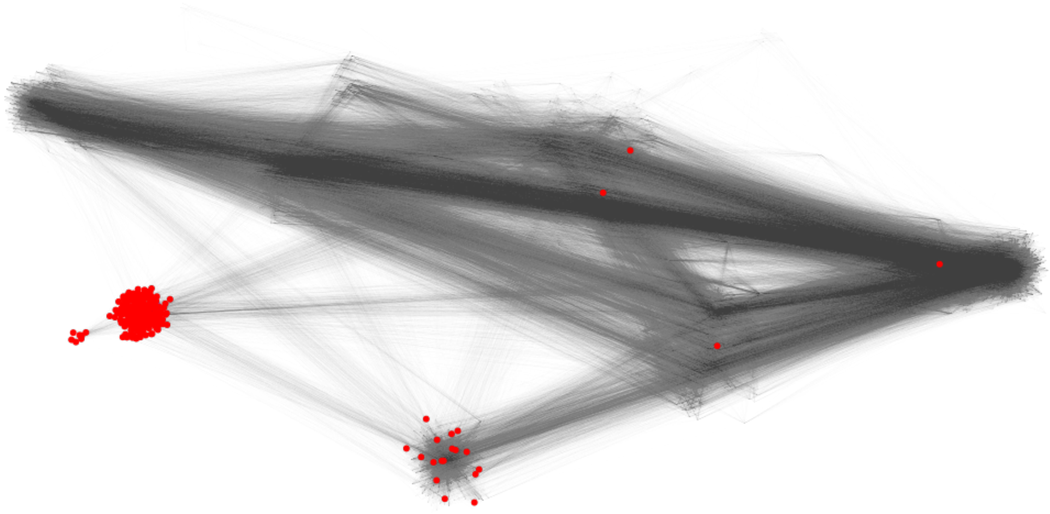
ℓ_1-regularized PageRank(best tuning)
Precision=0.73, Recall=0.94

Capacity Releasing Diffusion
Precision=0.93, Recall=0.94

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

Example on Facebook Johns Hopkins social network





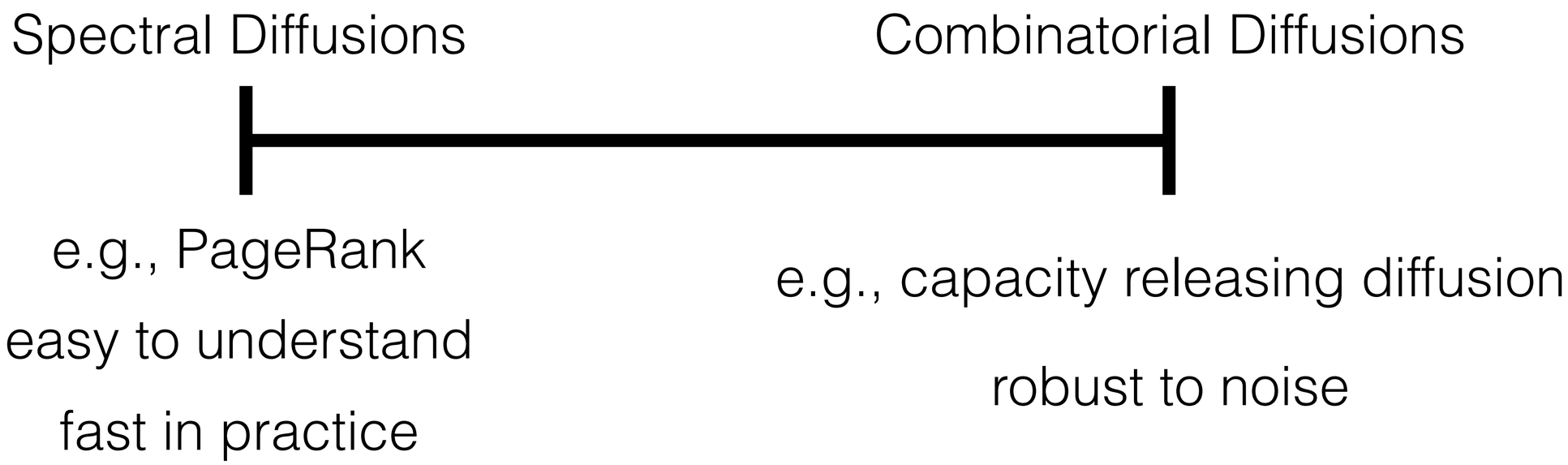
ℓ_1-regularized PageRank (best tuning)
Precision=0.71, Recall=0.91

Capacity Releasing Diffusion
Precision=0.87, Recall=0.94

D. Wang, K. Fountoulakis, M. Mahoney, S. Rao. Capacity Releasing Diffusion for Speed and Locality. ICML 2017

p-norm Flow Diffusions

Spectrum of methods



S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Spectrum of methods

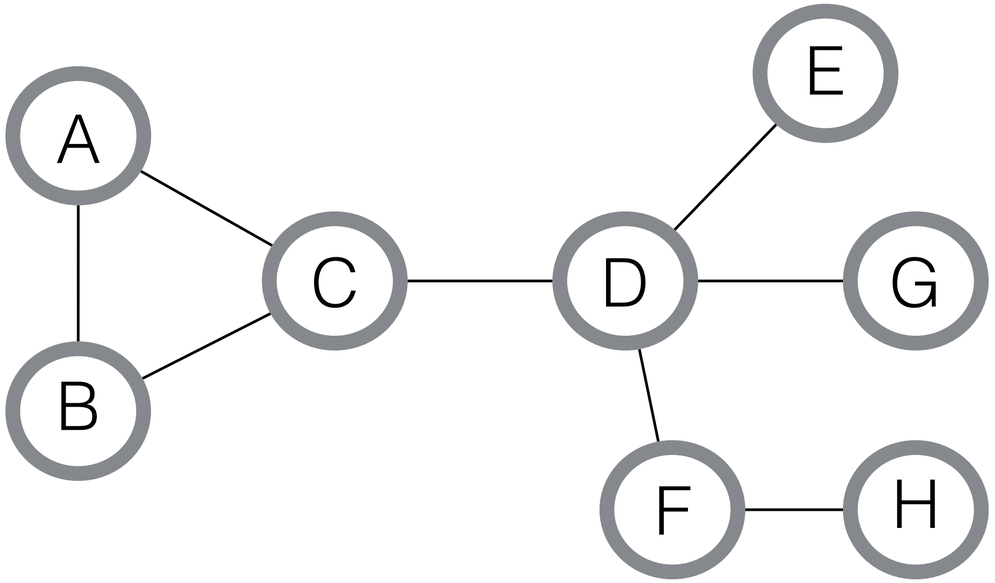
p-norm flow diffusion

- p-norm flow diffusion is a family of convex optimization problems that characterizes the trade-off between spectral and combinatorial diffusions.

- This allows us to define methods that are the best of both worlds.

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Some definitions - incidence matrix



Incidence matrix B

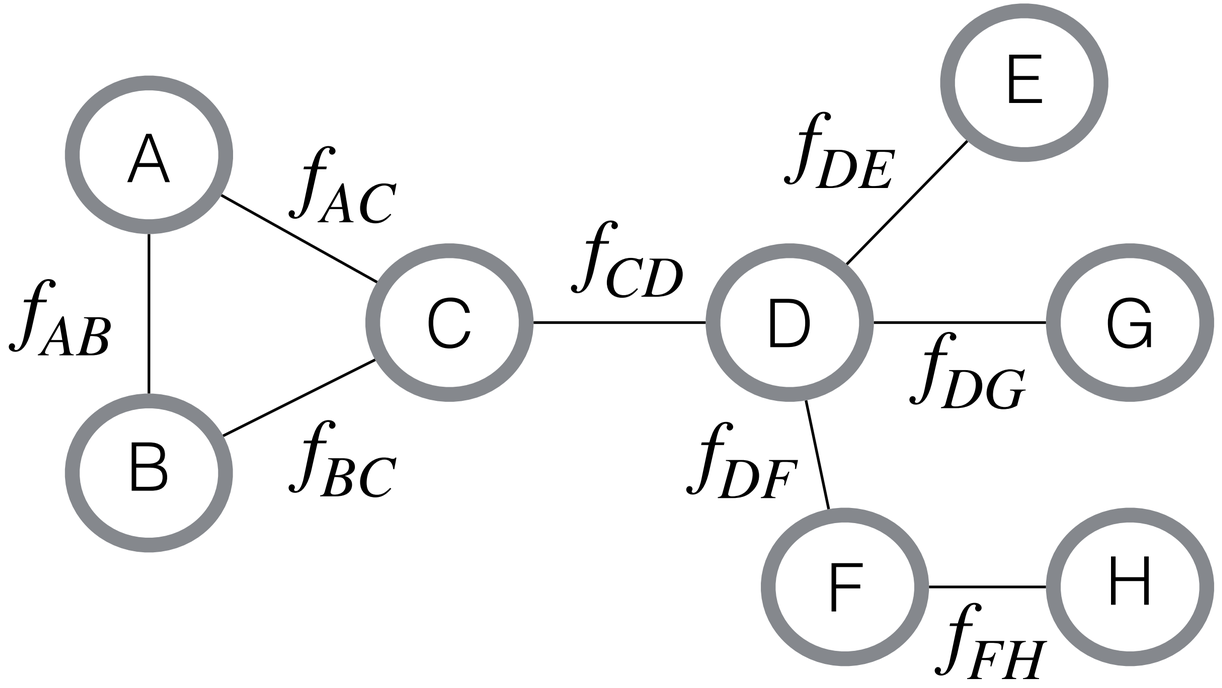
	A	B	C	D	E	F	G	H
A-B	1	-1						
A-C	1		-1					
B-C		1	-1					
C-D			1	-1				
D-E				1	-1			
D-F				1		-1		
D-G				1			-1	
F-H						1		-1

- Ordering of edges and direction is arbitrary

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Some definitions - flow variables

- Let f be a vector and each component of f corresponds to an edge, for example:



- The magnitude of f is the amount of flow that passes through an edge
- The sign of f is the direction of flow

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Some definitions - net flow

- Let Δ be a non-negative vector, each component of Δ indicates the initial mass at a node.

- $B^T f$ is a vector that captures the net flow on a node.

- $B^T f + \Delta$ indicates the net mass on every node.

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Node capacities

- We will require that each node has capacity equal to its degree d_i

- We will say that the initial mass Δ has been diffused, when the net mass on each node is less than its capacity:

$$\underbrace{B^T f + \Delta}_{\text{net mass per node}} \leq \underbrace{d}_{\text{capacity per node}}$$

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Diffusion as an optimization formulation

- Out of all possible flows f that satisfy the capacities we are interested in the one with minimum L_p norm, where $p \in [2,\infty)$.

$$\text{minimize } \|f\|_p$$
$$\text{subject to: } B^T f + \Delta \leq d$$

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Relation to other methods

- For $p = 2$ the dual of the 2-norm flow diffusion problem is

$$\text{minimize } \frac{1}{2}\|Bx\|_2^2 - x^T\Delta + \|Dx\|_1$$

- which is a regularized spectral problem, very similar ℓ_1-regularized PageRank.

- For $p \to \infty$ the dual of the ∞-norm flow diffusion problem is

$$\text{minimize } \|Bx\|_1 - x^T\Delta + \|Dx\|_1$$

- which is a regularized min-cut problem, very similar to the so-called flow-improve methods

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Rounding

- Sort the dual variables in descending order
- Output the prefix set with smallest conductance.

- In practice we solve the dual of the p-norm flow problem

$$\begin{aligned} \text{minimize } & -x^T\Delta + \|Dx\|_1 \\ \text{subject to: } & \|Bx\|_q \leq 1 \\ & x \geq 0 \end{aligned}$$

- So we have direct access to the dual variables

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

p-norm network flow diffusions - conductance guarantees

- **Theorem -** Let C be the target cluster with conductance $\Phi(C)$, if Δ is initialized inside C, and the input seed set sufficiently overlaps with C, then the output A satisfies

$$\Phi(A) \leq \mathcal{O}\left(\Phi(B)^{1-1/p}\right)$$

- Cheeger-type result for $p = 2$.
- Constant factor approximation when $p \to \infty$, similar to combinatorial diffusions.
- Smooth transition for general p values in between

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

p-norm network flow diffusions - algorithm

- Simple randomized coordinate descent

- Running time $$\mathcal{O}\left(\frac{|\Delta|}{\gamma}\left(\frac{|\Delta|}{\epsilon}\right)^{1-2/p}\log\frac{1}{\epsilon}\right)$$

- $|\Delta|$ represents the magnitude of the initial mass.
- γ is the strong convexity parameter of the dual problem.
- ϵ is the required accuracy

- $p = 2$ gives the usual running time for spectral methods $\tilde{\mathcal{O}}(|\Delta|)$
- $p \to \infty$ gives the usual running time for combinatorial methods $\tilde{\mathcal{O}}(|\Delta|^2/\epsilon)$

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

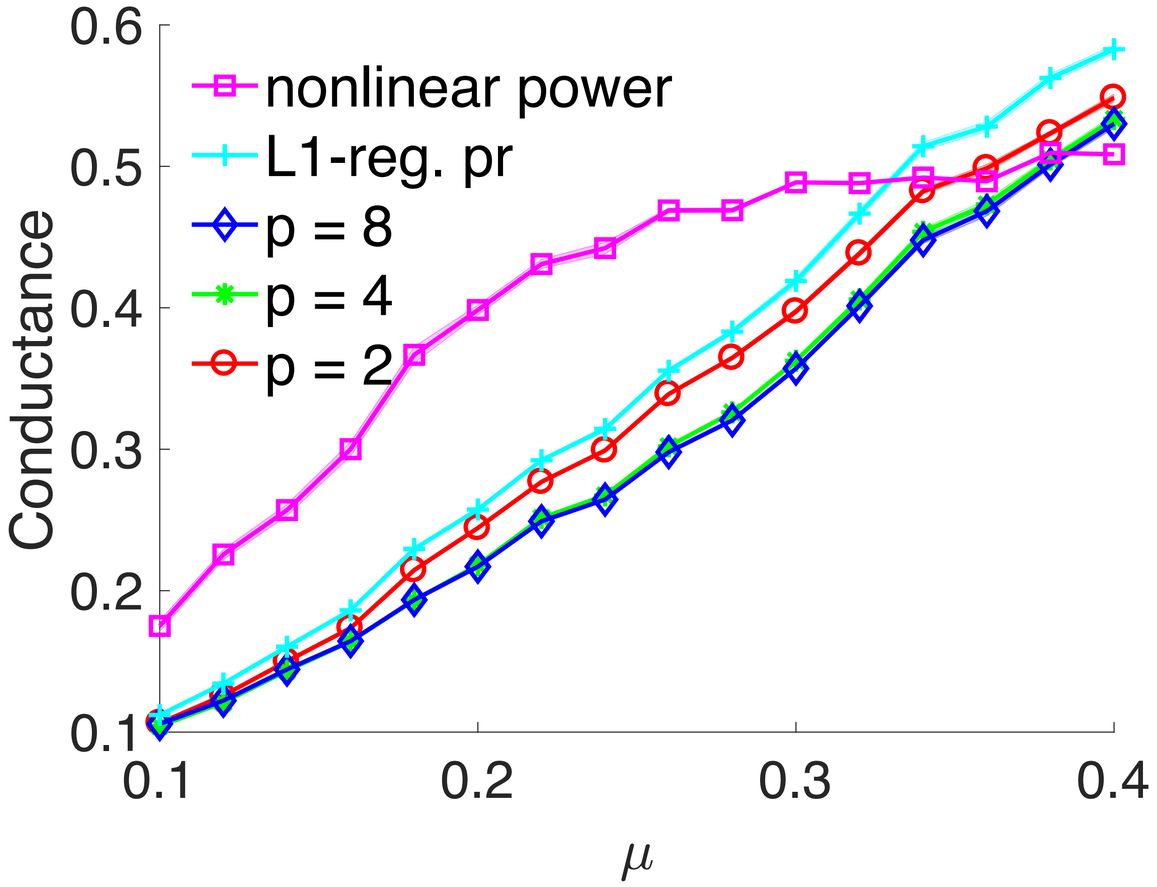
p-norm network flow diffusions - summary

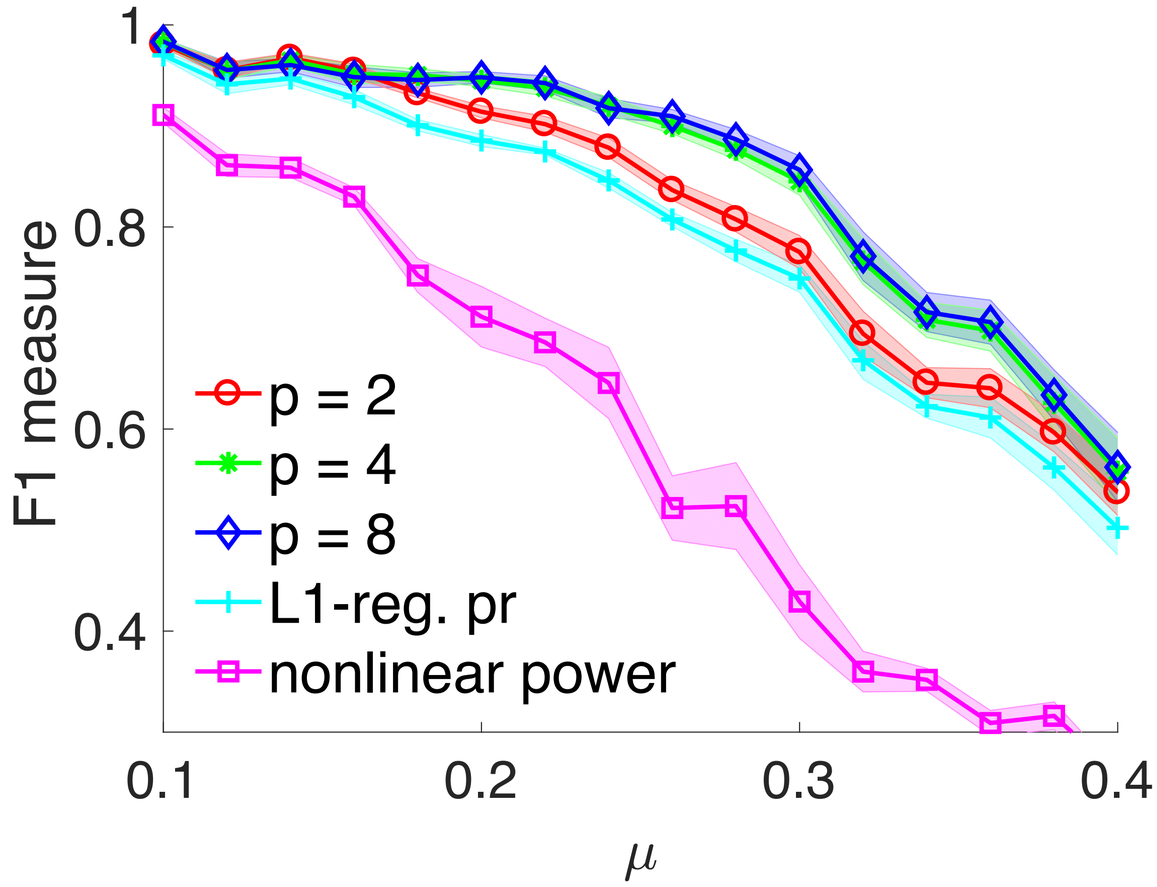
- There is a trade-off between quality of output and running time
- The larger p is the better the output with respect to conductance.
- However, the larger p is the more the running time for solving problem.
- In practice, small values of $p \in [2,8]$ gives the best of both worlds.

S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

Performance

- LFR synthetics model, basically a stochastic block model
- μ is a parameter that controls noise, the higher the more noise.



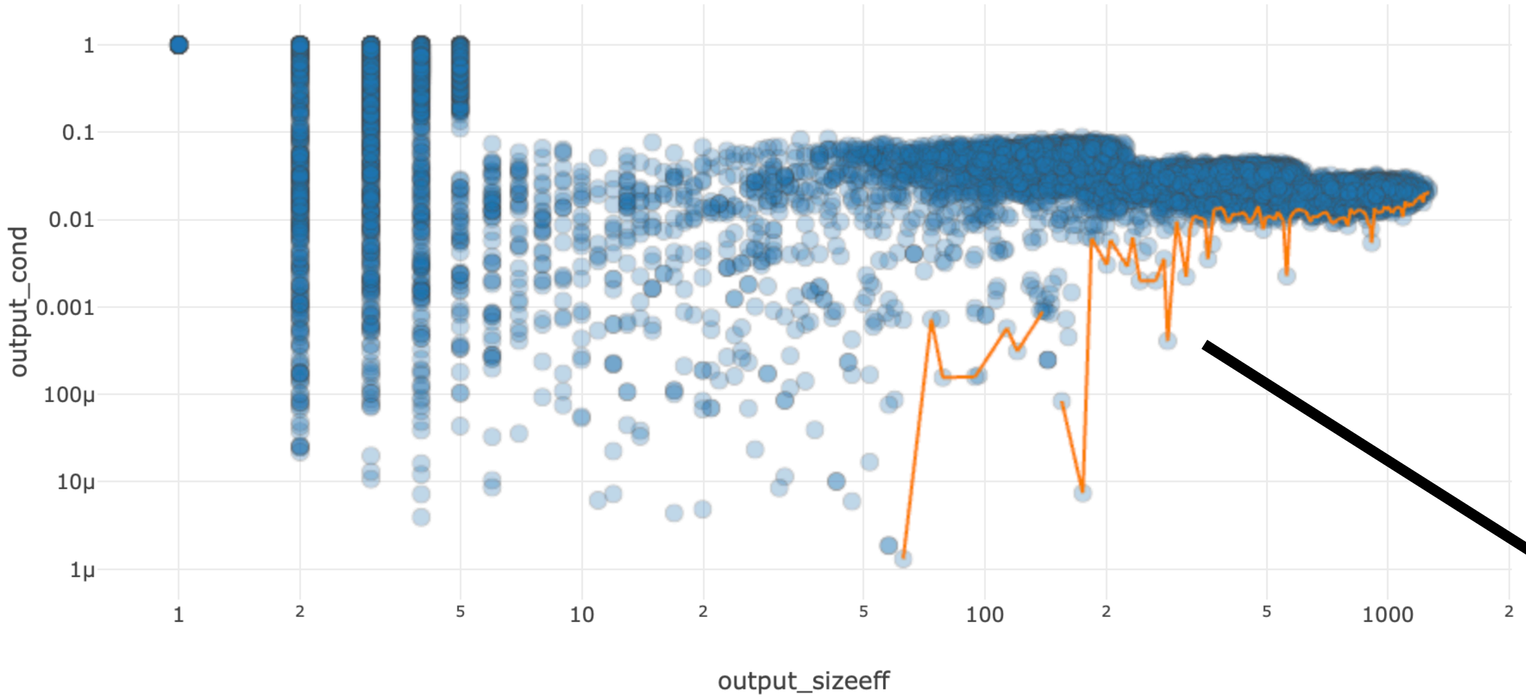


S. Yang, D. Wang, K. Fountoulakis. p-Norm Flow Diffusion for Local Graph Clustering.

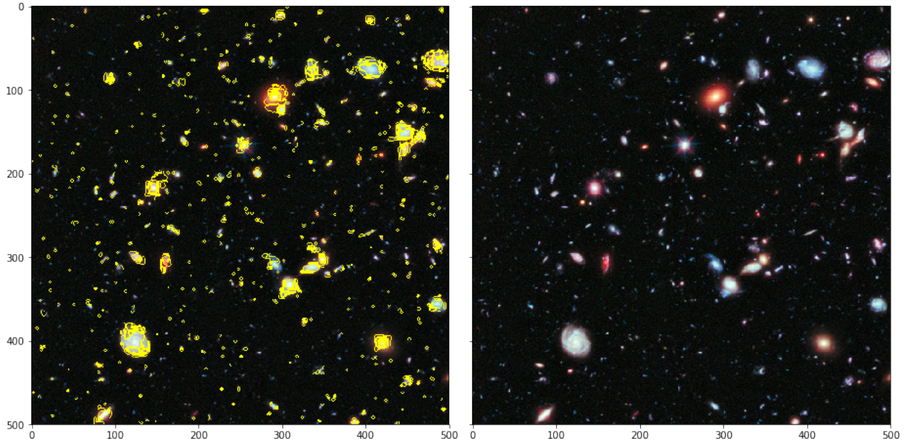
Local to Global Applications: Network Community Profiles, Node embeddings, Graph Visualization, Semi-Supervised Learning

(no theory ☹, preliminary work)

Network Community Profiles



Clusters with smallest conductance correspond to galaxies



Node embeddings

- **Goal:** Represent a node with a low dimensional vector.
- We use node embeddings for graph visualization, semi-supervised learning and graph partitioning.

Types of node embeddings

- Global embeddings
- Local embeddings, i.e., spectral and combinatorial

Global embeddings

- Compute the Laplacian matrix $L = D - A$
- Compute k non-trivial eigenvectors of L
- Stuck the eigenvectors as columns of a $n \times k$ matrix U.
- Each row of U is a vector representation (node embedding) of a node.

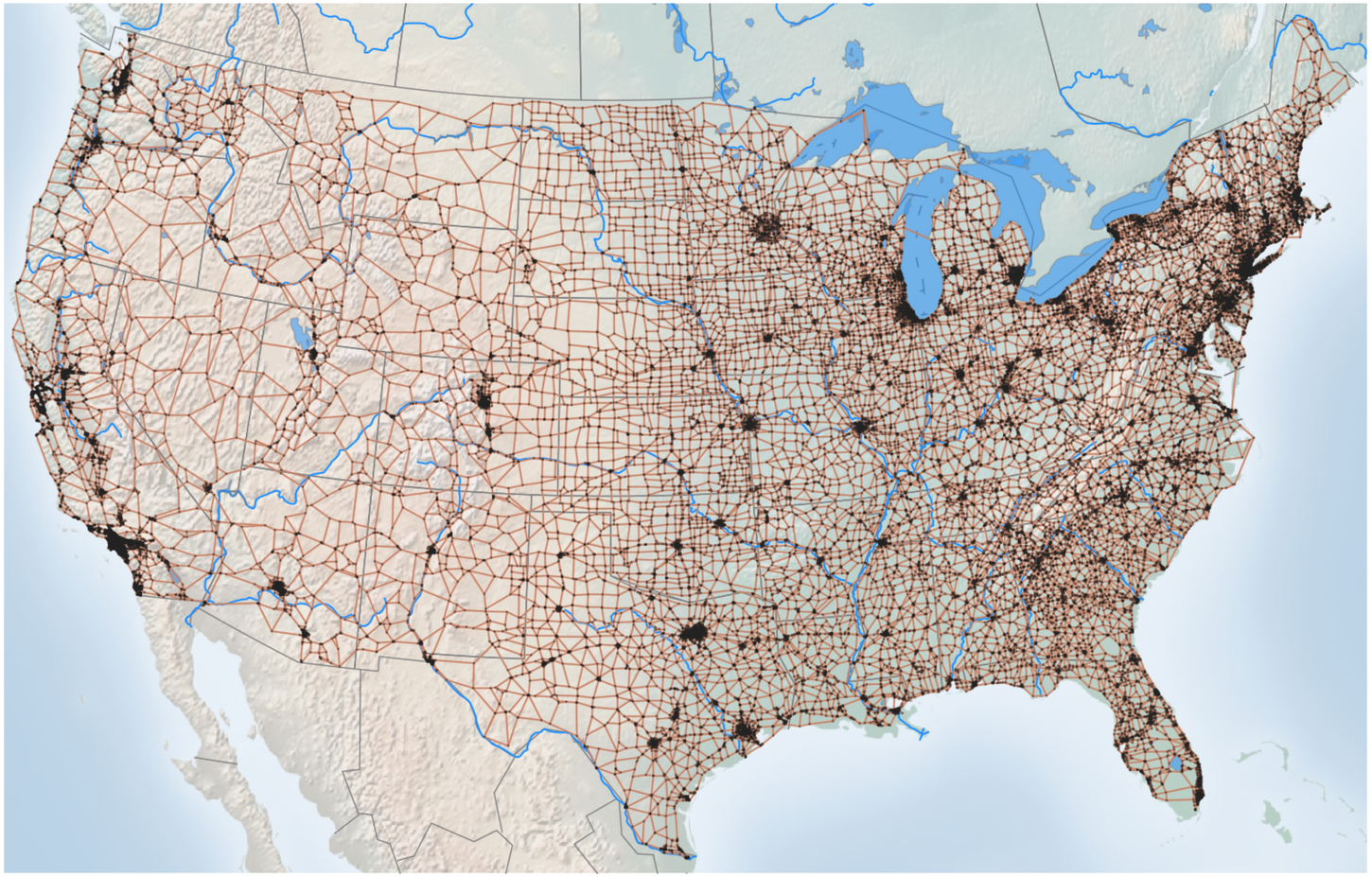
Local spectral embeddings

- Choose randomly N seed sets
- For each seed set run a local ***spectral*** algorithm.
- Stuck eigenvectors as columns of a $n \times N$ matrix X.
- Compute k principal left singular vectors of X
- Stuck the singular vectors as columns of a $n \times k$ matrix U.
- Each row of U is a vector representation (node embedding) of a node.

Local flow embeddings

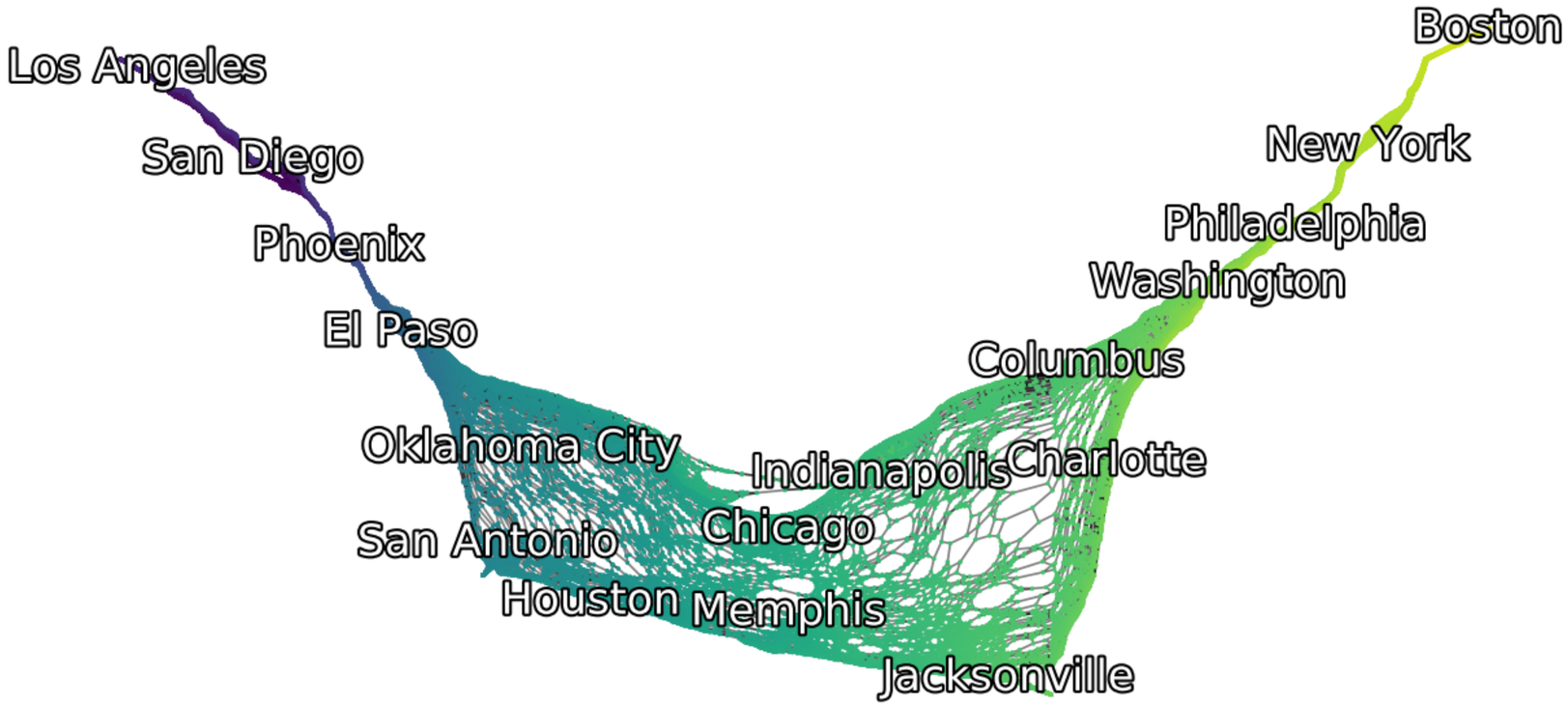
- Choose randomly N seed sets
- For each seed set run a local ***flow*** algorithm
- Stuck eigenvectors as columns of a $n \times N$ matrix X.
- Compute k principal left singular vectors of X
- Stuck the singular vectors as columns of a $n \times k$ matrix U.
- Each row of U is a vector representation (node embedding) of a node.

Graph visualization - US highway network



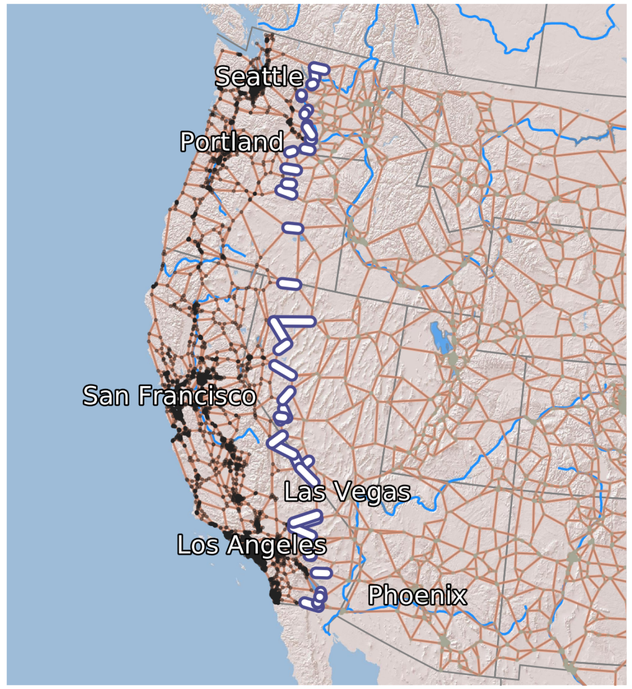
- Edges represent naturally funded highways, and nodes represent intersections.
- Mostly toy-graph for demonstration purposes

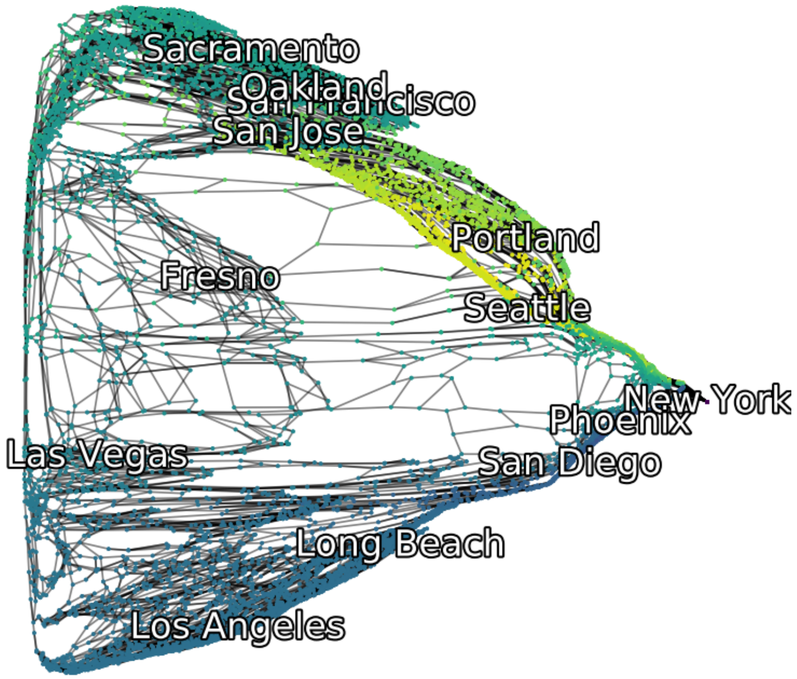
Graph visualization - global embeddings

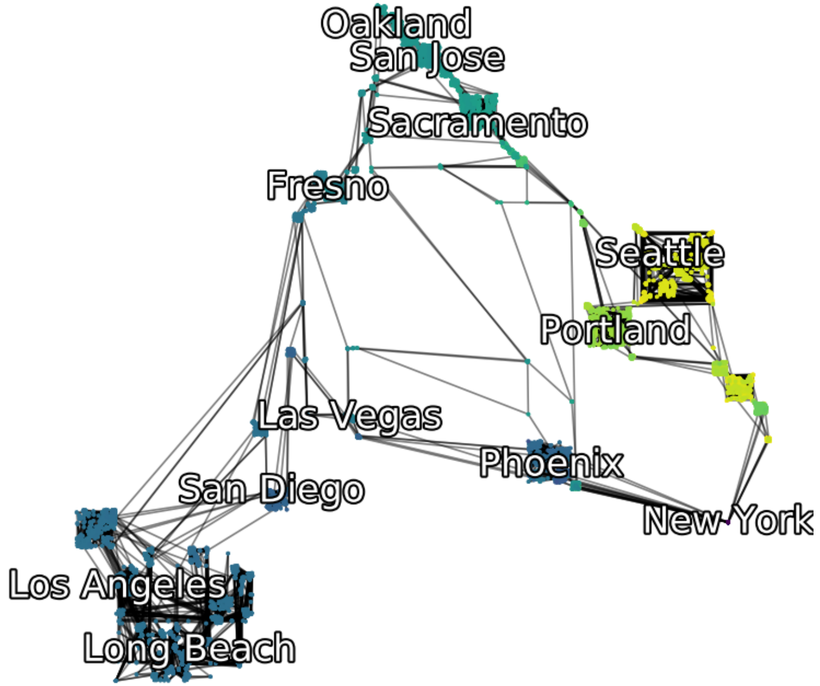


- Color shows true longitude
- Global embeddings seem to correlate with longitude
- But, **compresses major regions** of the northeastern US (Washington, New York, Boston) as well as the Western US (Los Angeles, San Diego, Phoenix).

Graph visualization - local embeddings







map | Local spectral embeddings | Local flow embeddings

- With global embeddings Western US (Los Angeles, San Diego, Phoenix) was quite compressed.
- Local embeddings help in de-compressing the region.
- Local spectral and flow embeddings seem to be qualitatively different.

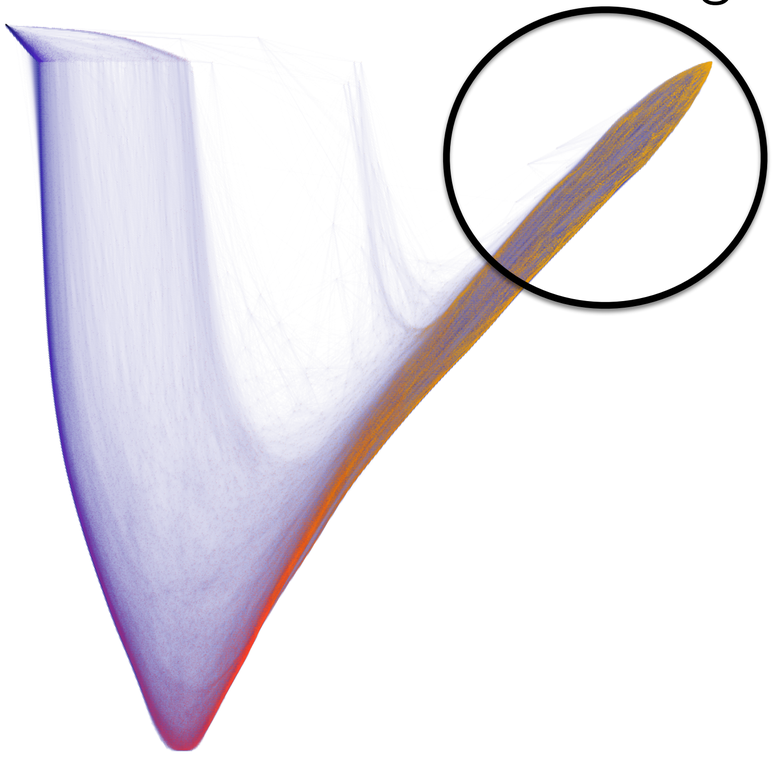
Main Galaxy Sample data

- Each node is a galaxy
- Edges represent distance among galaxies
- The distance is determined by measuring the distance of the emission spectra of two galaxies
- There are 517182 galaxies (nodes) and each galaxy has 4 neighbor galaxies (edges)

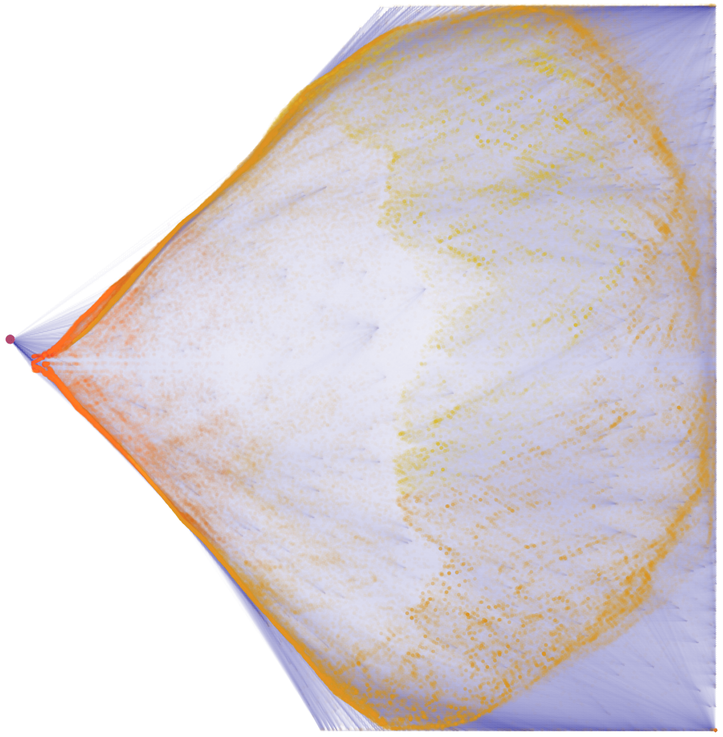
Mapping the similarities of spectra: global and local approaches to sdss galaxies. The Astrophysical Journal. 2016.

Local spectral and flow embeddings - Main Galaxy Sample data

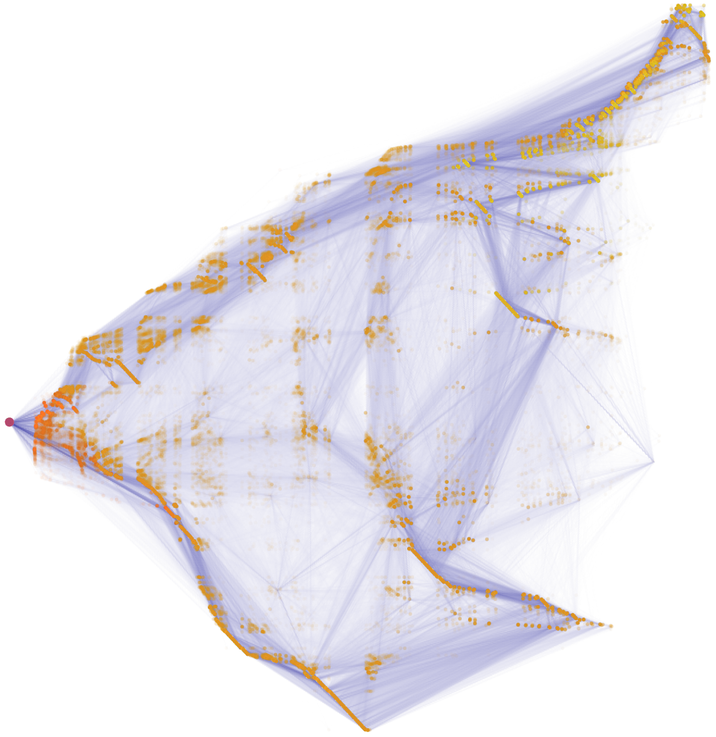
Zoom-in this
dense region



Global embedding

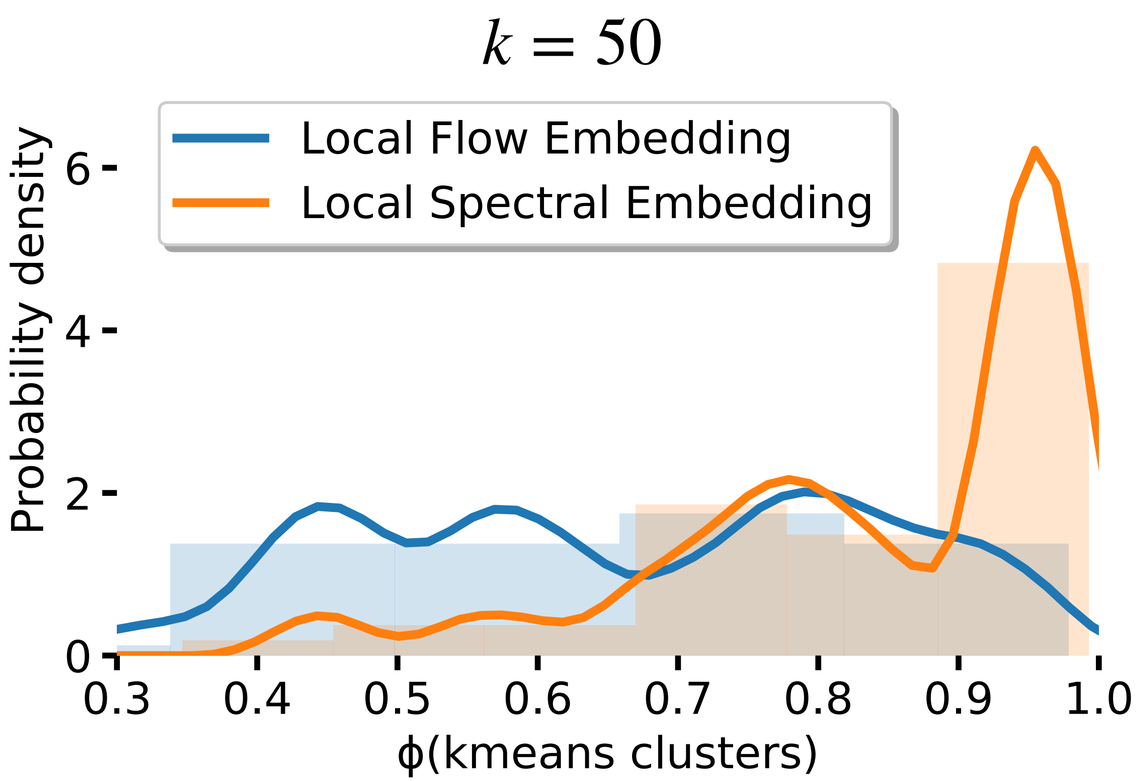


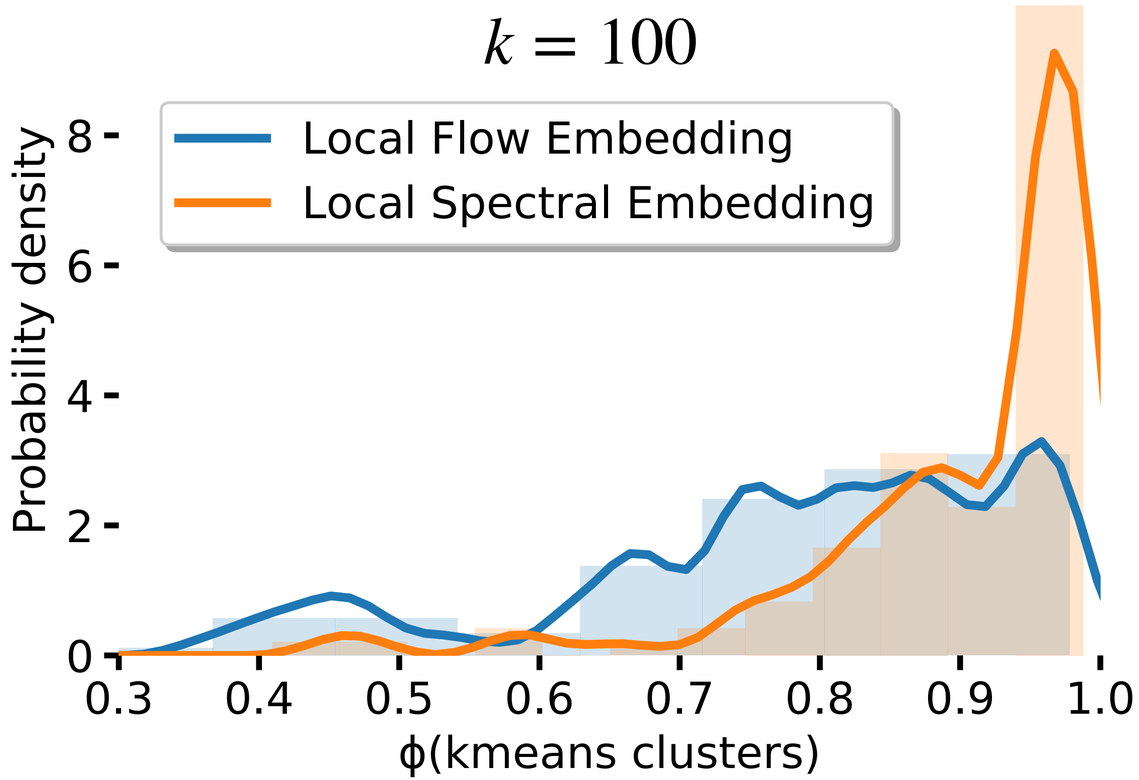
Local spectral



Local flow

Local spectral and flow embeddings - Main Galaxy Sample data





- Structural differences in visualization also translate to clusters with smaller conductance.

Semi-supervised learning

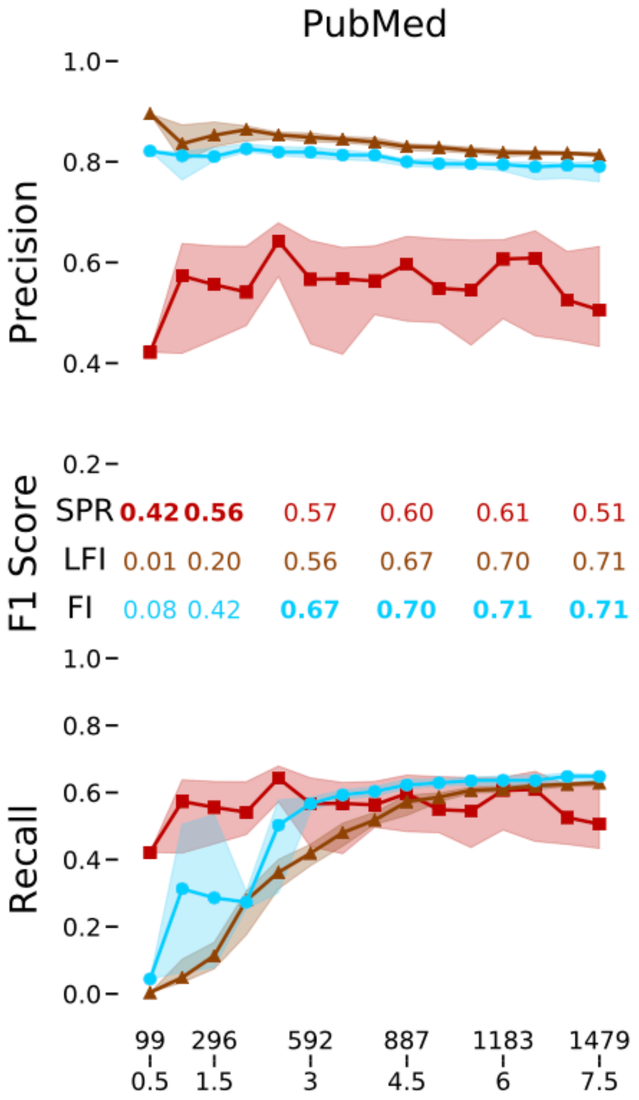
Problem

- Infer unknown labels for all nodes, when given a few nodes with known labels.
- We assume that the graph edges represent a high likelihood of sharing a label.

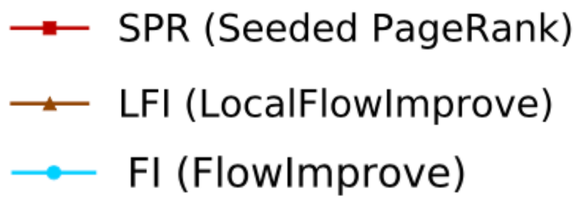
Algorithm

- For each class, we randomly select a small subset of nodes, and we fix the labels of these nodes as known.
- We then run a spectral or a flow method where this set of nodes is the reference. This gives one spectral or flow vector per class.
- For each unlabelled node we look at the corresponding coordinate in the vectors and we give it the label that corresponds to the class with the highest value.

Semi-supervised learning



True labels included in seeds



Info about the data

- PubMed is a citation network. 19717 scientific publications about diabetes with 44338 citation links.
- By construction of the graph, articles about one type of diabetes cite others about the same type more often.

Software

LocalGraphClustering on **GitHub** 

Thank you!