Graph Mining and Multi-Relational Learning: Tools and Applications

Shobeir Fakhraei Amazon

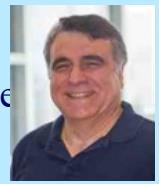
Christos Faloutsos Amazon & CMU

Bird's eye view

Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking									
1.1' Link Prediction									
1.2 Comm. Detection									
1.3 Anomaly Detection									
1.4 Propagation									

Part 1: Plain Graphs

Part 2: Complex Graphs

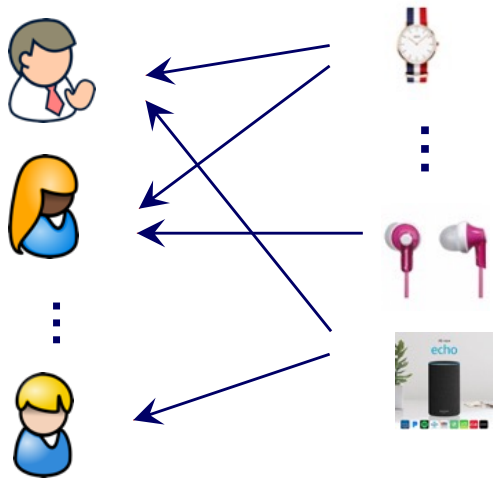


Bird's eye view

- Introduction - motivation
- Part#1: (plain) Graphs
- Part#2: MRL, Tensors etc
- Conclusions

e-commerce examples

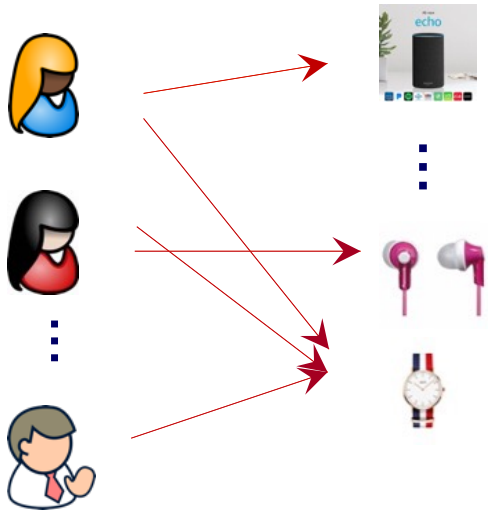
Who-buys-what ←



e-commerce examples

Who-buys-what ←

Who-sells-what →

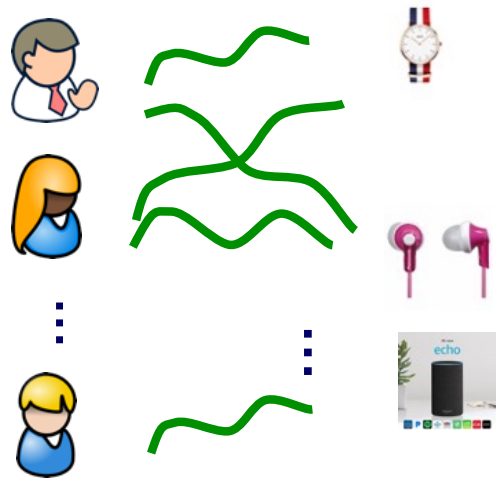


e-commerce examples

Who-buys-what

Who-sells-what

Who-reviews-what

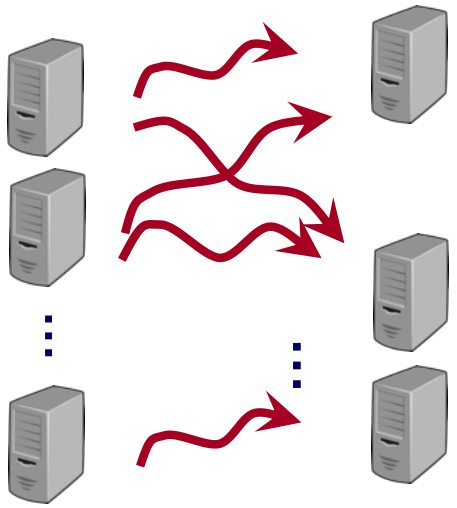


Cyber-security

Who-buys-what

Who-sells-what

Who-reviews-what



Which_machine - connects_to - what

…

<subject> related-to <object> : graph

Examples on complex graphs?

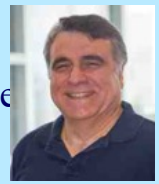
(all the previous examples, are on 'plain' graphs)

Bird's eye view

Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking									
1.1' Link Prediction									
1.2 Comm. Detection									
1.3 Anomaly Detection									
1.4 Propagation									

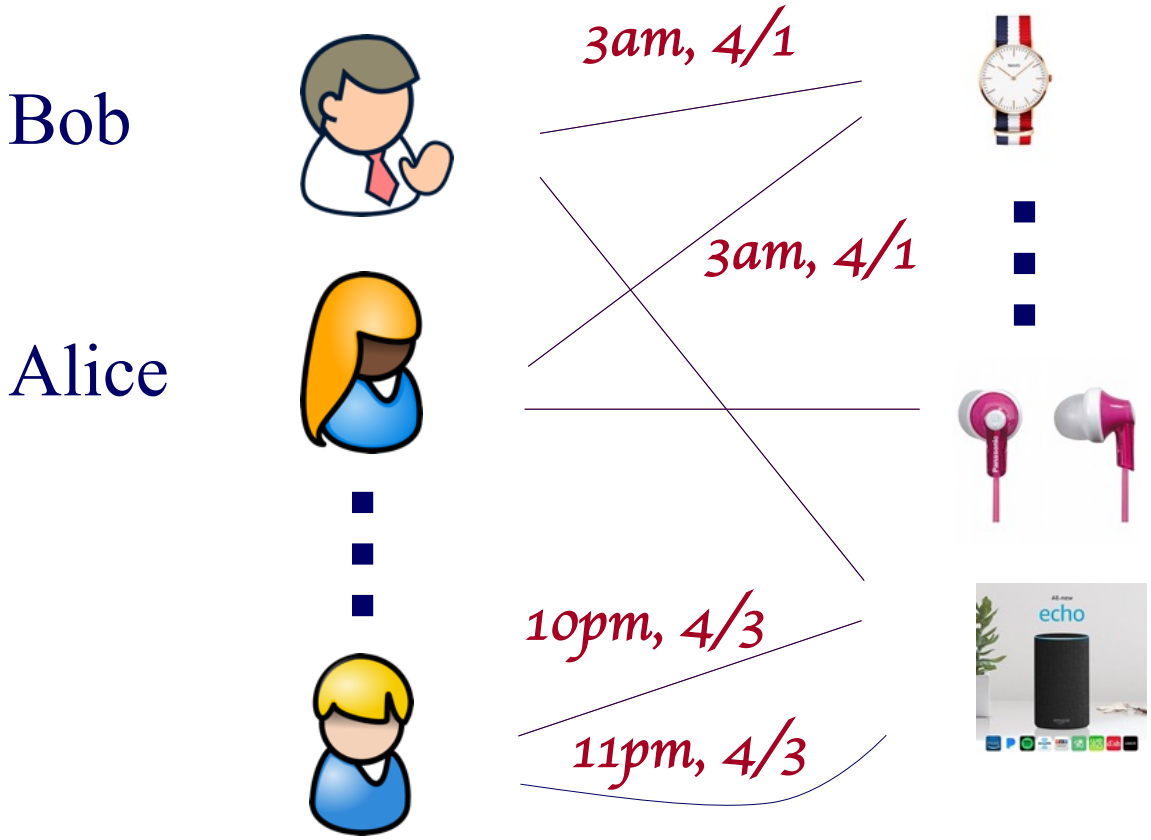
Part 1: Plain Graphs

Part 2: Complex Graphs



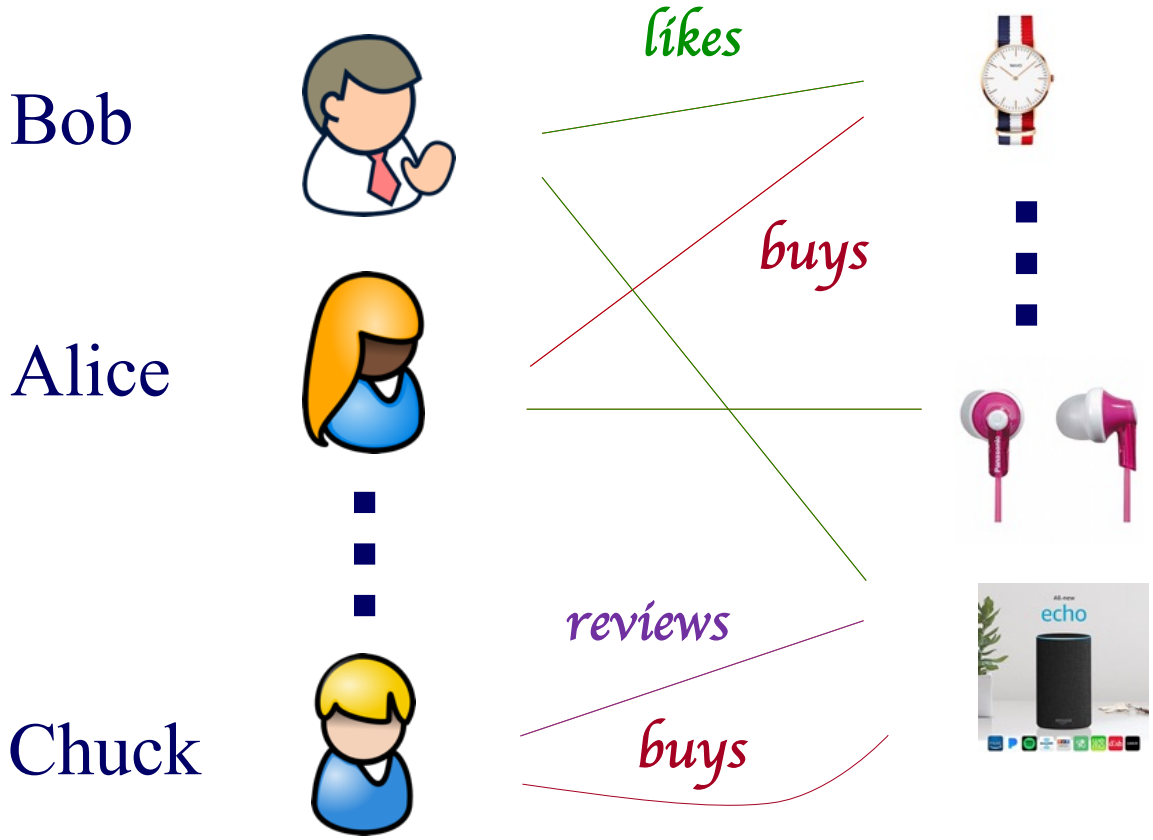
Complex, e.g., time-evolving graphs

- What is 'normal'? suspicious? Groups?



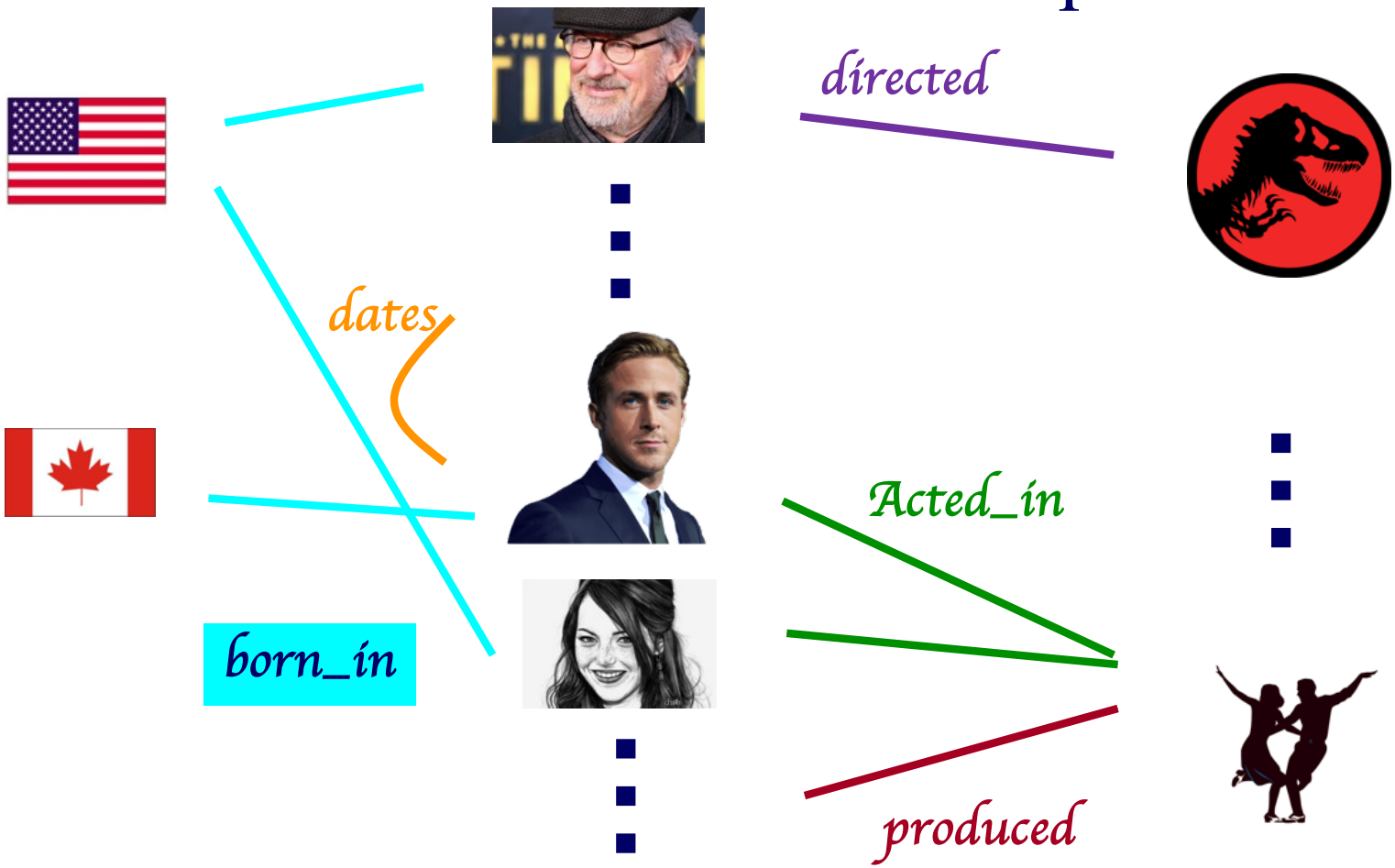
Complex, e.g., MultiView Graph

- What is 'normal'? suspicious? Groups?



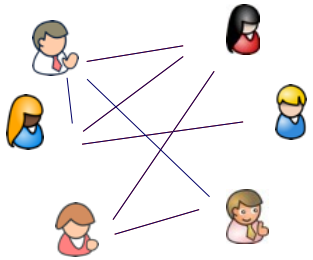
Complex, e.g., Knowledge Graph

- What is 'normal'? Forecast? Spot errors?



Definitions

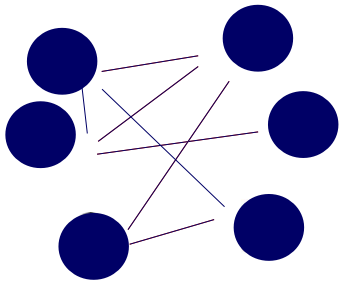
plain



Complex

Definitions

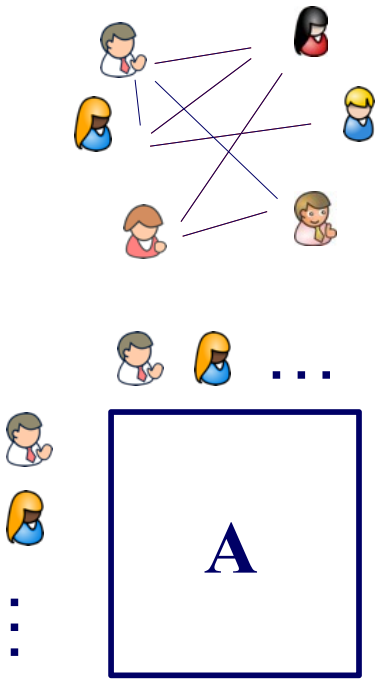
plain



Complex

Definitions

plain



Matrix

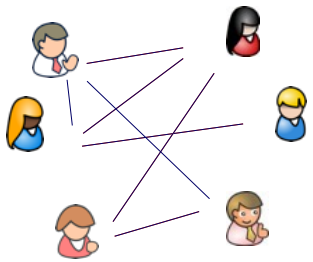
Complex

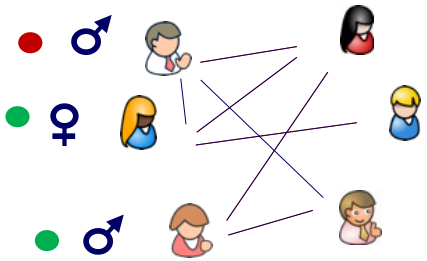
Definitions

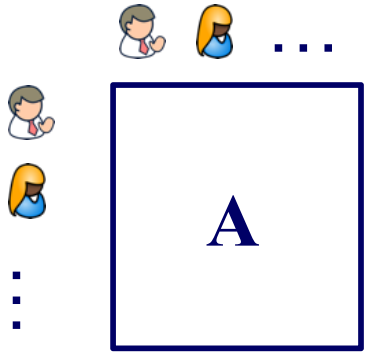
plain

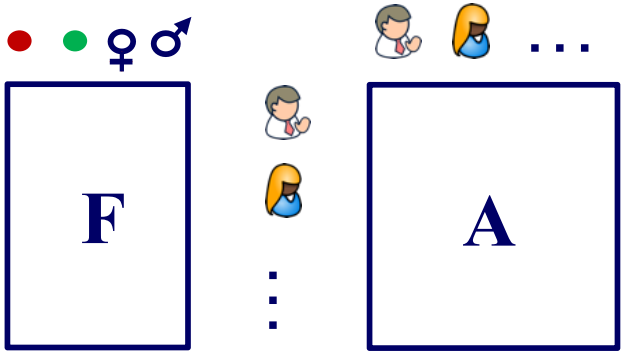
Complex

Node-attr.









Matrix

Coupled Matrices

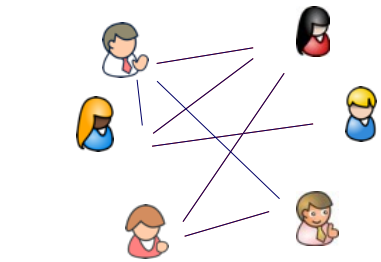
Definitions

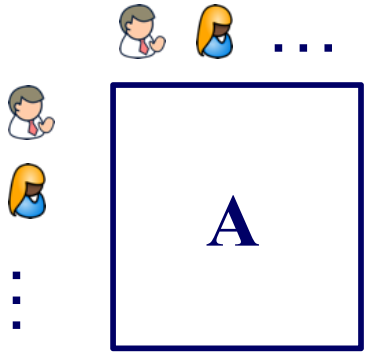
plain

Complex

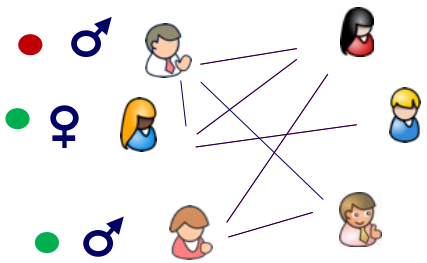
Node-attr.

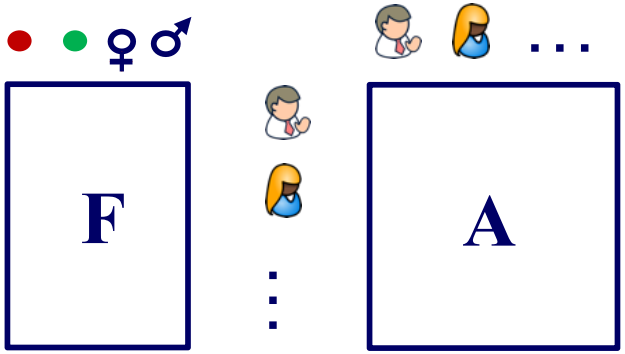
Edge-attr.



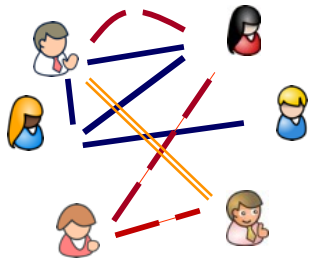


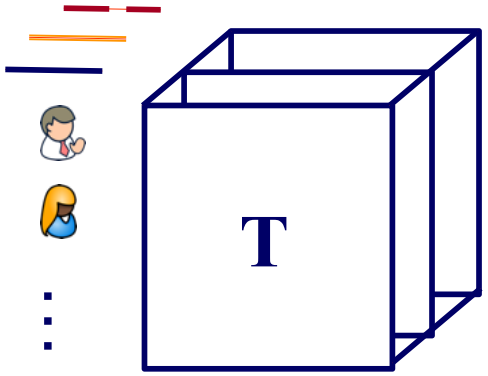
Matrix





Coupled Matrices



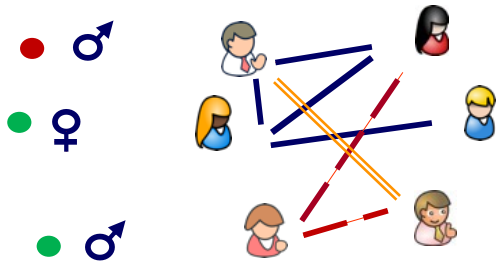


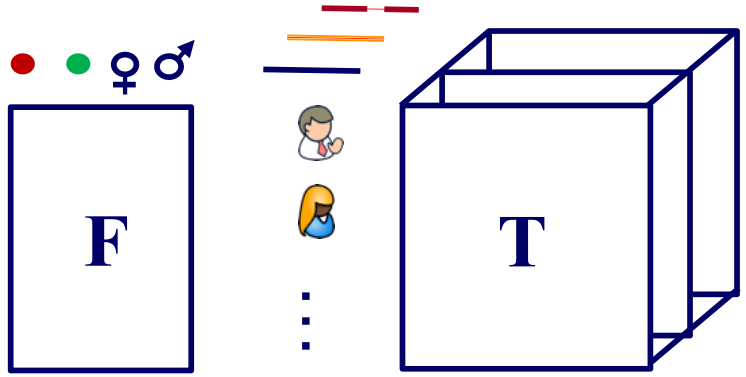
Tensor

Definitions

'Complex' include any combination:

- Edge AND node attributes
- Timestamps
- Locations
- …



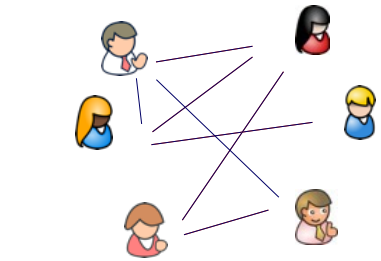


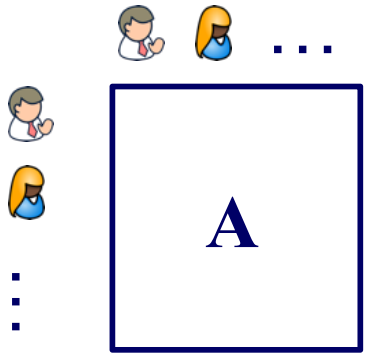
Coupled Matrix-Tensor

Definitions

PART 1

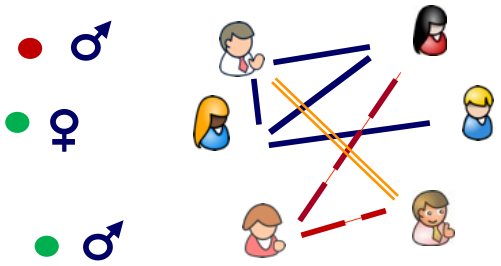
Plain graphs

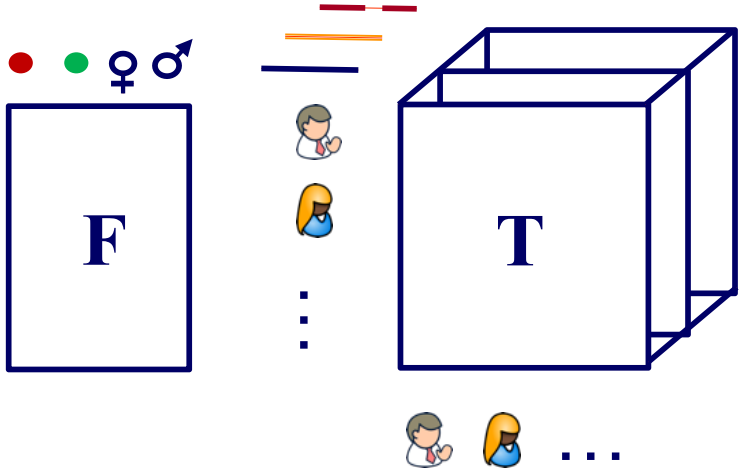




PART 2

Complex graphs

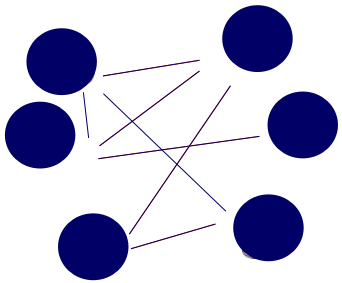


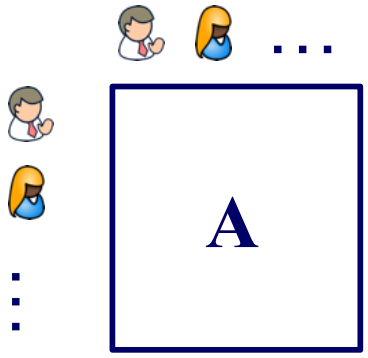


Definitions

PART 1

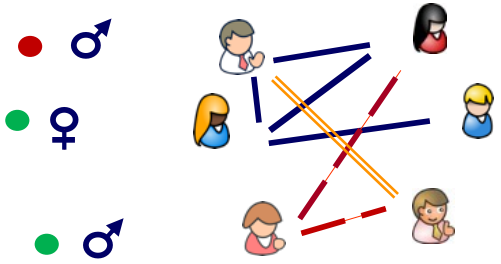
Plain graphs

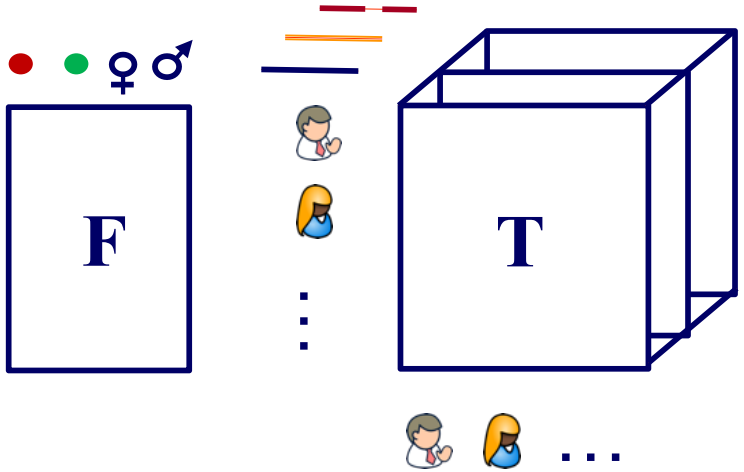




PART 2

Complex graphs



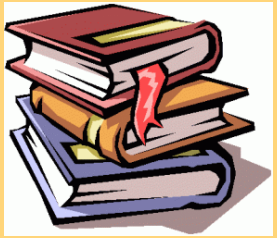


‘Recipe’ Structure:

- Problem definition

- Short answer/solution

- LONG answer – details



- Conclusion/short-answer

‘Recipe’ Structure:

- Problem definition

- Short answer/solution

- LONG answer – details

- Conclusion/short-answer

NOT covered here

- Deep Learning / GNN
- See, eg., www.dgl.ai/
  - w/ tutorials and s/w
  - from aws colleagues

Bird's eye view

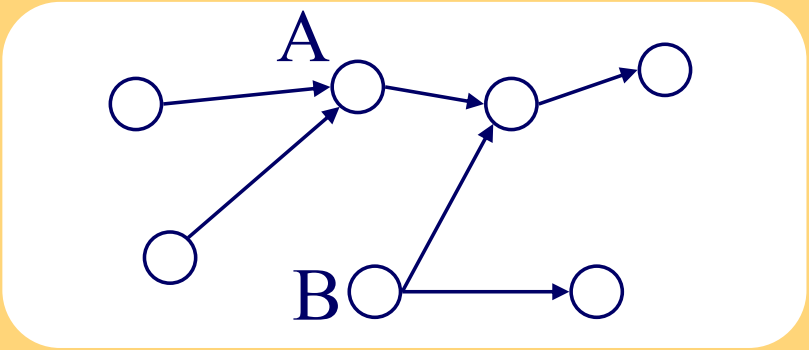
Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking									
1.1' Link Prediction									
1.2 Comm. Detection									
1.3 Anomaly Detection									
1.4 Propagation									

Part 1: Plain Graphs

Part 2: Complex Graphs

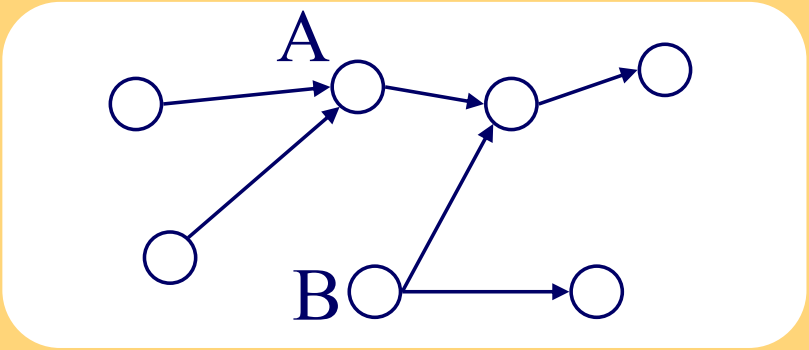
Node importance - Motivation:

- Given a graph (eg., web pages containing the desirable query word)
- Q1: Which node is the most important?
- Q2: How close is node ‘A’ to node ‘B’?



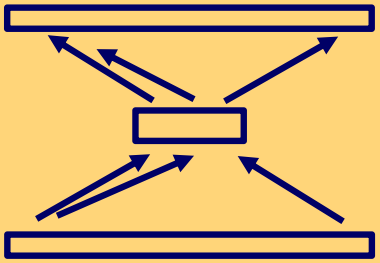
Node importance - Motivation:

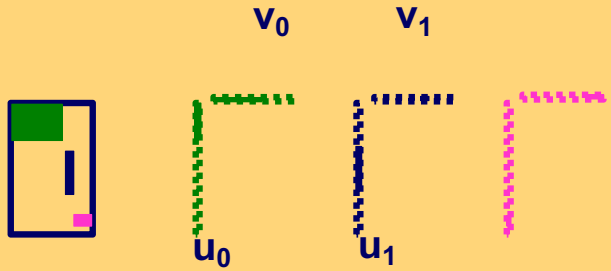
- Given a graph (eg., web pages containing the desirable query word)
- Q1: Which node is the most important?
  - **PageRank (PR = RWR)**, HITS (SVD)
- Q2: How close is node 'A' to node 'B'?
  - Personalized P.R. (PPR)



SVD properties (Singular Value Decomposition)

- ✓ Hidden/latent variable detection
- ✓ Compute node importance (HITS)
- ✓ Block detection
- ✓ Dimensionality reduction
- ✓ Embedding (linear)
  - SVD is a special case of 'deep neural net'

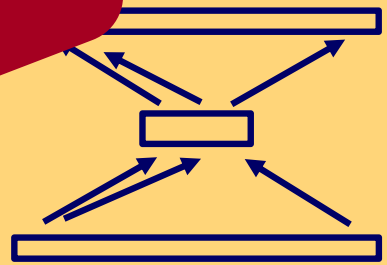


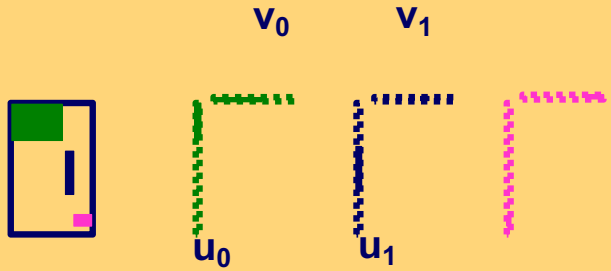


SVD properties

- ✓ Hidden/latent variable detection
- ✓ Compute node importance
- ✓ Block detection
- ✓ Dimensionality reduction
- ✓ Embedding (spectral)
  - SVD is a special case of 'deep neural net'

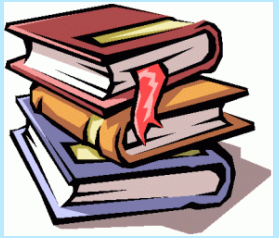
Matrix? SVD!

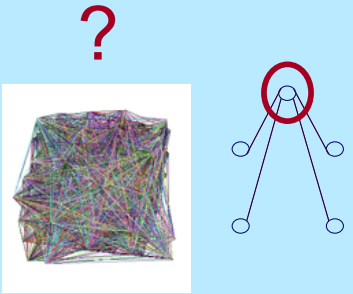




Bird's eye view

- Introduction – Motivation
- Part#1: (simple) Graphs
  - P1.1: node importance
    - PageRank and Personalized PR
    - HITS
    - SVD





PageRank (google)

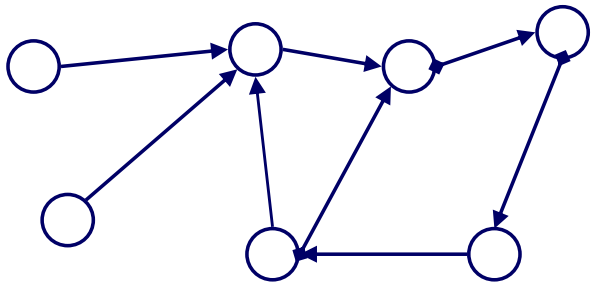
Larry Page

Sergey Brin

- Brin, Sergey and Lawrence Page (1998). *Anatomy of a Large-Scale Hypertextual Web Search Engine*. 7th Intl World Wide Web Conf.
- Page, Brin, Motwani, and Winograd (1999). *The PageRank citation ranking: Bringing order to the web*. Technical Report

Problem: PageRank

Given a directed graph, find its most interesting/central node

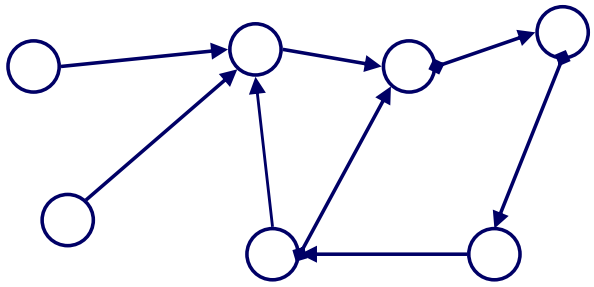


A node is important,
if its parents are important
(recursive, but OK!)

Problem: PageRank - solution

Given a directed graph, find its most interesting/central node

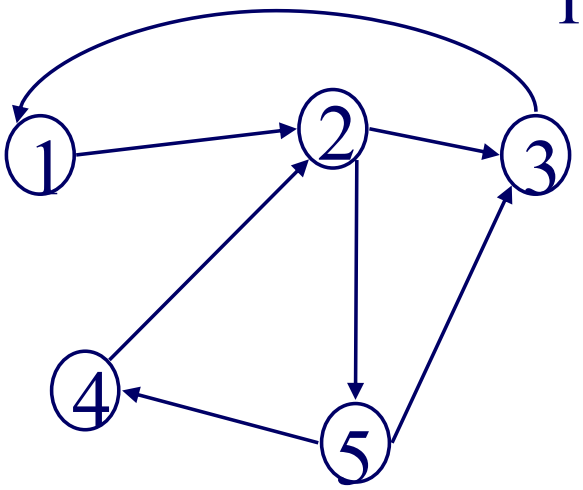
Proposed solution: Random walk; spot most 'popular' node (-> steady state prob. (ssp))



A node high ssp,
if its parents have high ssp
(recursive, but OK!)

(Simplified) PageRank algorithm

- Let **A** be the adjacency matrix;
- let **B** be the transition matrix: transpose, column-normalized - then



From / To

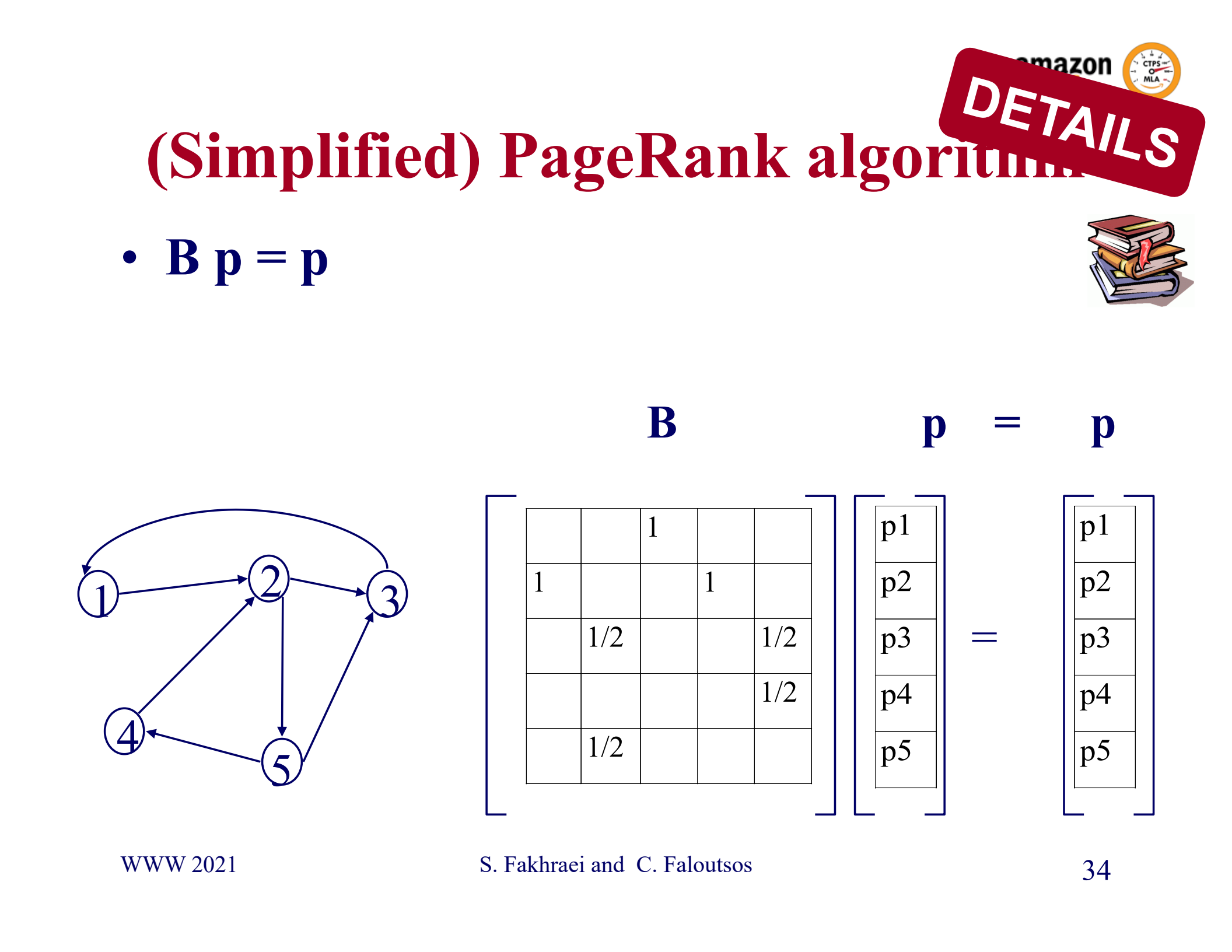
B

		1		
1			1	
	1/2			1/2
				1/2
	1/2			

p1
p2
p3
p4
p5

=

p1
p2
p3
p4
p5



(Simplified) PageRank algorithm

- **B p = p**

B **p** = **p**

		1		
1			1	
	1/2			1/2
				1/2
	1/2			

p1
p2
p3
p4
p5

=

p1
p2
p3
p4
p5

Definitions

A Adjacency matrix (from-to)

D Degree matrix = (diag (d1, d2, …, dn))

B Transition matrix: to-from, column normalized

$$\mathbf{B} = \mathbf{A}^T \mathbf{D}^{-1}$$

(Simplified) PageRank algorithm

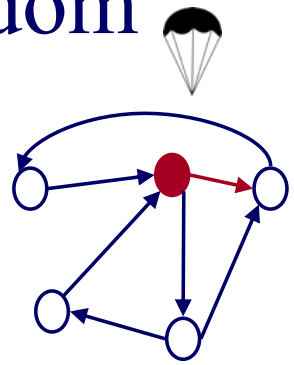
- **B p** = 1 * **p**
- thus, **p** is the **eigenvector** that corresponds to the highest eigenvalue (=1, since the matrix is column-normalized)
- Why does such a **p** exist?
  - **p** exists if **B** is nxn, nonnegative, irreducible [Perron–Frobenius theorem]

(Simplified) PageRank algorithm

- In short: imagine a particle randomly moving along the edges
- compute its steady-state probabilities (ssp)

Full version of algo: with occasional random jumps

Why? To make the matrix irreducible

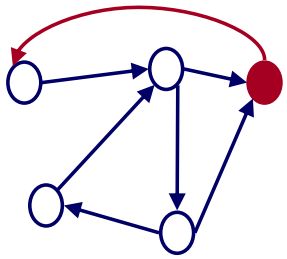


(Simplified) PageRank algorithm

- In short: imagine a particle randomly moving along the edges
- compute its steady-state probabilities (ssp)

Full version of algo: with occasional random jumps

Why? To make the matrix irreducible

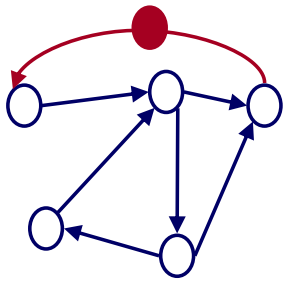


(Simplified) PageRank algorithm

- In short: imagine a particle randomly moving along the edges
- compute its steady-state probabilities (ssp)

Full version of algo: with occasional random jumps

Why? To make the matrix irreducible

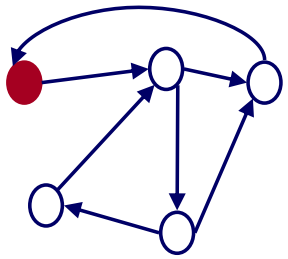


(Simplified) PageRank algorithm

- In short: imagine a particle randomly moving along the edges
- compute its steady-state probabilities (ssp)

Full version of algo: with occasional random jumps

Why? To make the matrix irreducible

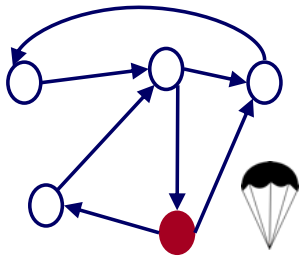


(Simplified) PageRank algorithm

- In short: imagine a particle randomly moving along the edges
- compute its steady-state probabilities (ssp)

Full version of algo: with occasional random jumps

Why? To make the matrix irreducible



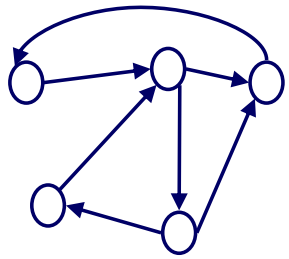
(Simplified) PageRank algorithm

- In short: imagine a particle randomly moving along the edges
- compute its steady-state probabilities (ssp)

PageRank = PR

= Random Walk with Restarts = RWR

= Random surfer

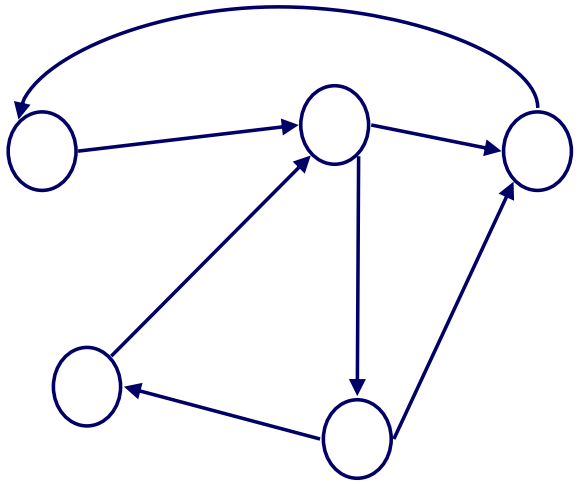


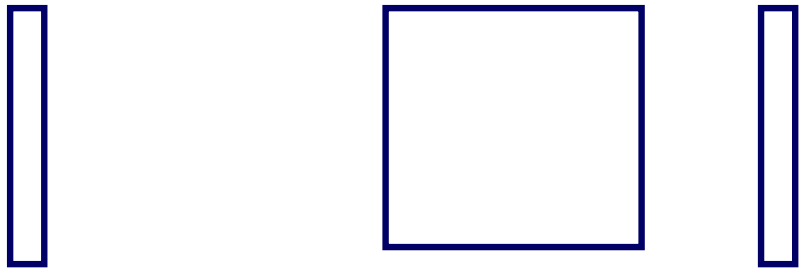
Full Algorithm

- With probability *1-c*, fly-out to a random node
- Then, we have

$$\mathbf{p} = c\,\mathbf{B}\,\mathbf{p} + (1-c)/n\,\mathbf{1} \Rightarrow$$

$$\mathbf{p} = (1-c)/n\,\left[\mathbf{I} - c\,\mathbf{B}\right]^{-1}\,\mathbf{1}$$



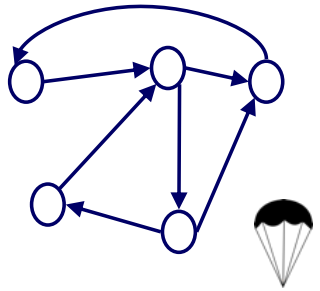


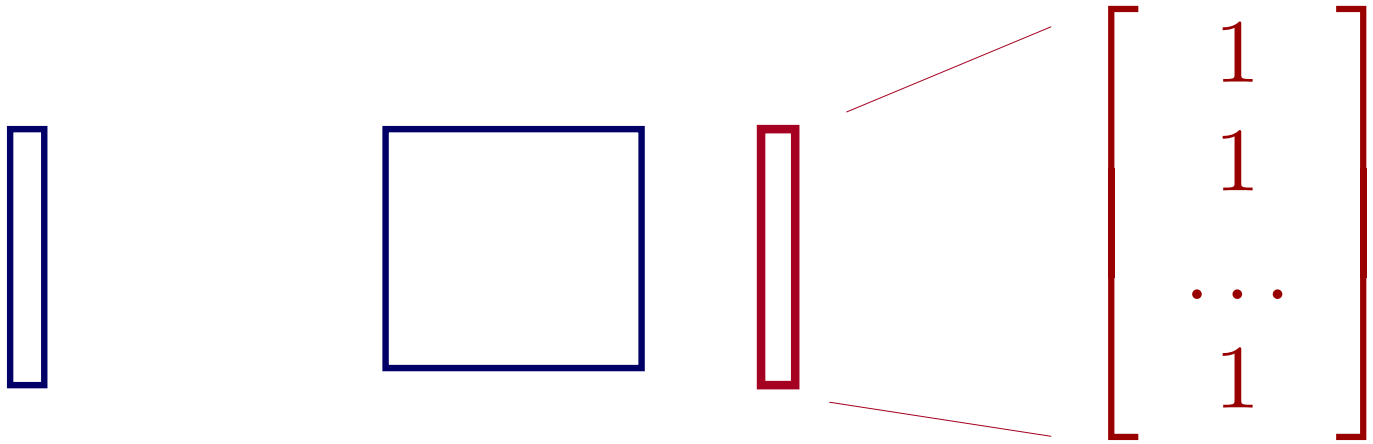
Full Algorithm

- With probability *1-c*, fly-out to a random node
- Then, we have

$$\mathbf{p} = c\,\mathbf{B}\,\mathbf{p} + (1-c)/n\,\mathbf{1} \Rightarrow$$

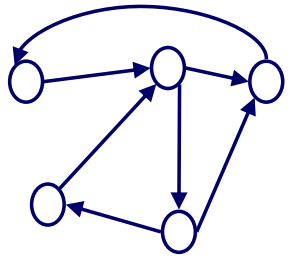
$$\mathbf{p} = (1-c)/n\,\left[\mathbf{I} - c\,\mathbf{B}\right]^{-1}\,\mathbf{1}$$





Notice:

- pageRank ~ in-degree
- (and HITS, also: ~ in-degree)



Bird's eye view

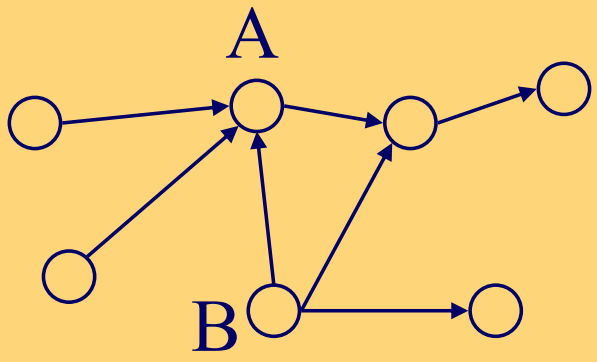
Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction									
1.2 Comm. Detection									
1.3 Anomaly Detection									
1.4 Propagation									

Part 1: Plain Graphs

Part 2: Complex Graphs

Node importance - Motivation:

- Given a graph (eg., web pages containing the desirable query word)
- Q1: Which node is the most important?
- Q2: How close is node ‘A’ to node ‘B’?

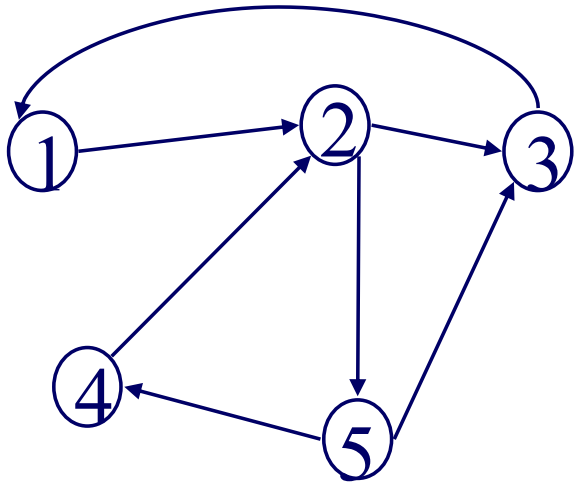


Personalized P.R.

- Taher H. Haveliwala. 2002. *Topic-sensitive PageRank.* (WWW '02). 517-526. http://dx.doi.org/10.1145/511446.511513

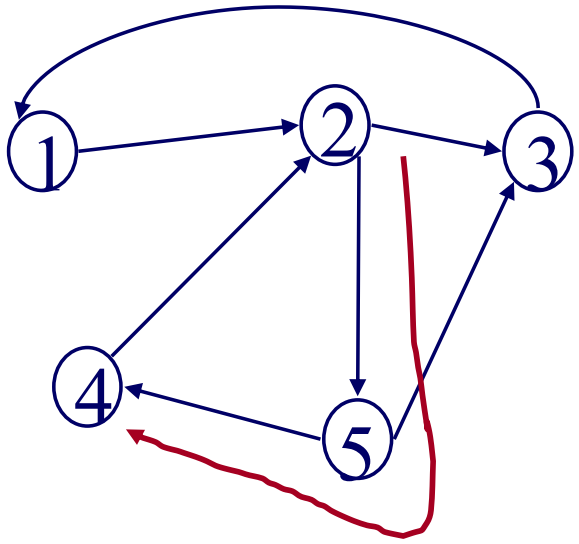
Extension: Personalized P.R.

- How close is '4' to '2'?
- (or: if I like page/node '2', what else would you recommend?)



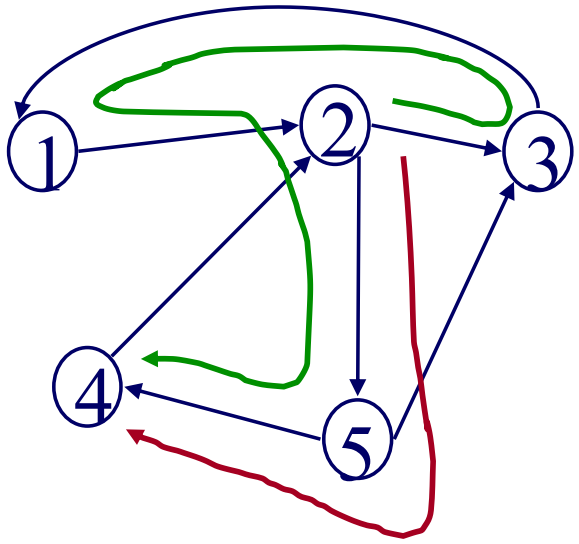
Extension: Personalized P.R.

- How close is ‘4’ to ‘2’?
- (or: if I like page/node ‘2’, what else would you recommend?)



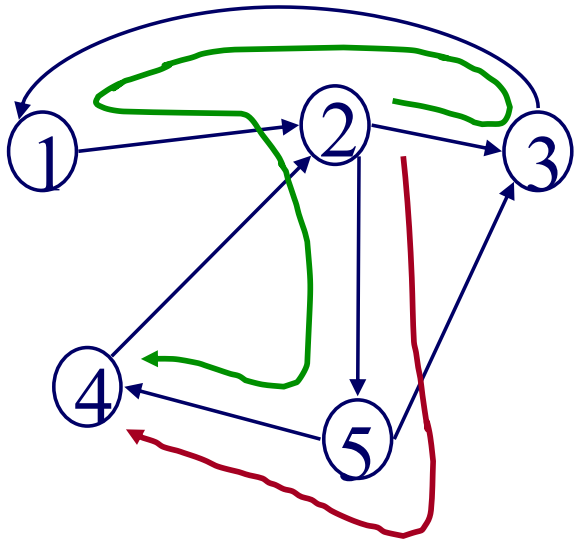
Extension: Personalized P.R.

- How close is ‘4’ to ‘2’?
- (or: if I like page/node ‘2’, what else would you recommend?)



Extension: Personalized P.R.

- How close is ‘4’ to ‘2’?
- (or: if I like page/node ‘2’, what else would you recommend?)



High score (A -> B) if

- Many
- Short
- Heavy

paths A->B

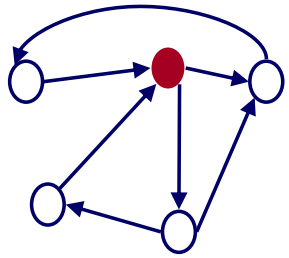
Extension: Personalized P.R

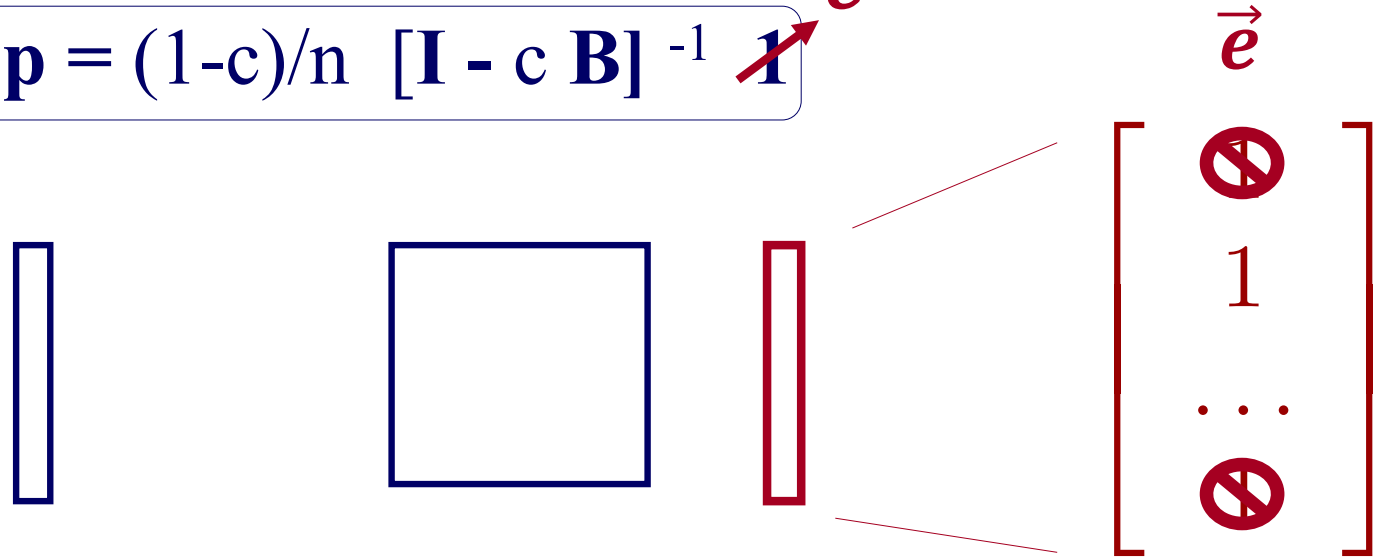
your favorite

- With probability *1-c*, fly-out to ~~a random~~ node(s)
- Then, we have

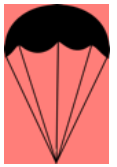
$$\mathbf{p} = c\,\mathbf{B}\,\mathbf{p} + (1-c)/n\,\vec{e} \Rightarrow$$

$$\mathbf{p} = (1-c)/n\,[\mathbf{I} - c\,\mathbf{B}]^{-1}\,\vec{e}$$

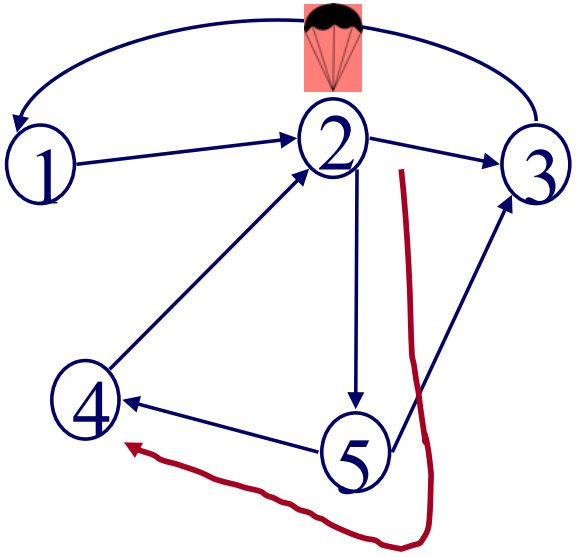




Extension: Personalized P.R.

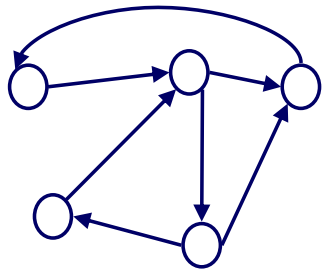


- How close is ‘4’ to ‘2’?
- A: compute Personalized P.R. of ‘4’, restarting from ‘2’



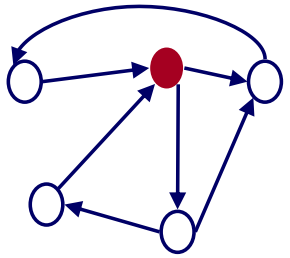
Extension: Personalized P.R.

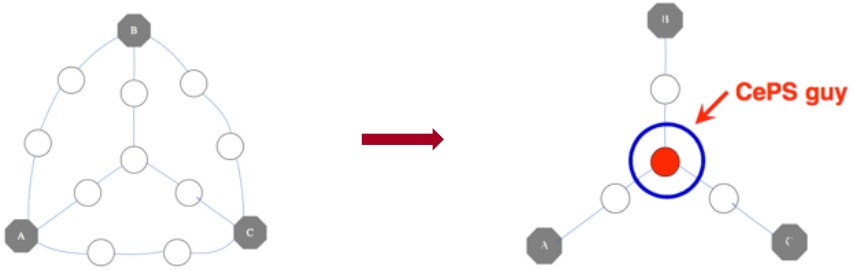
- How close is ‘4’ to ‘2’?
- A: compute Personalized P.R. of ‘4’, restarting from ‘2’ – Related to
  - ‘escape’ probability
  - ‘round trip’ probability
  - …



Applications of node proximity

- Recommendation
- Link prediction
- ‘Center Piece Subgraphs’
- …





Fast Algorithms for Querying and Mining Large Graphs
Hanghang Tong, PhD dissertation, CMU, 2009. TR: CMU-ML-09-112.

Bird's eye view

Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction		👍							
1.2 Comm. Detection									
1.3 Anomaly Detection									
1.4 Propagation									

Part 1: Plain Graphs

Part 2: Complex Graphs

Bird's eye view

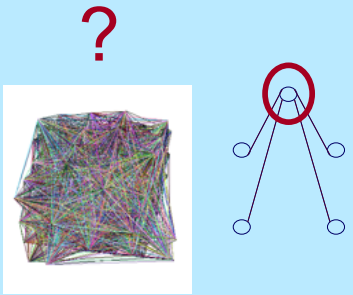
Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction		👍							
1.2 Comm. Detection									
1.3 Anomaly Detection									
1.4 Propagation									

Part 1: Plain Graphs

Part 2: Complex Graphs

Bird's eye view

- Introduction – Motivation
- Part#1: (simple) Graphs
  - P1.1: node importance
    - PageRank and Personalized PR
    - HITS
    - SVD (Singular Value Decomposition)

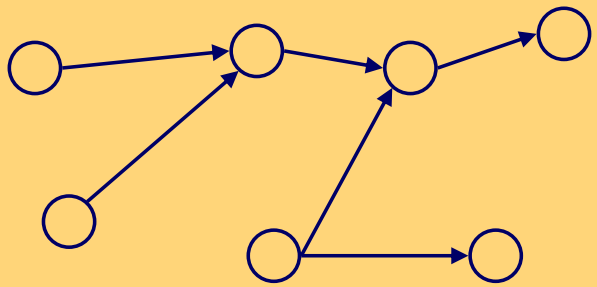


Kleinberg's algo (HITS)

Kleinberg, Jon (1998). *Authoritative sources in a hyperlinked environment*. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms.

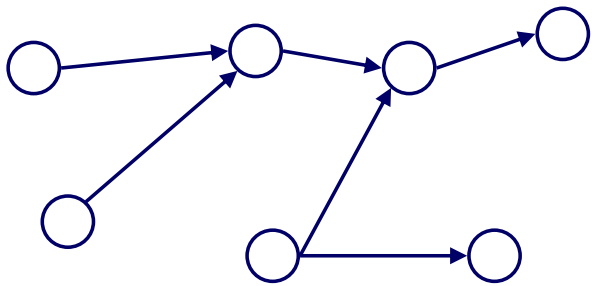
Recall: problem dfn

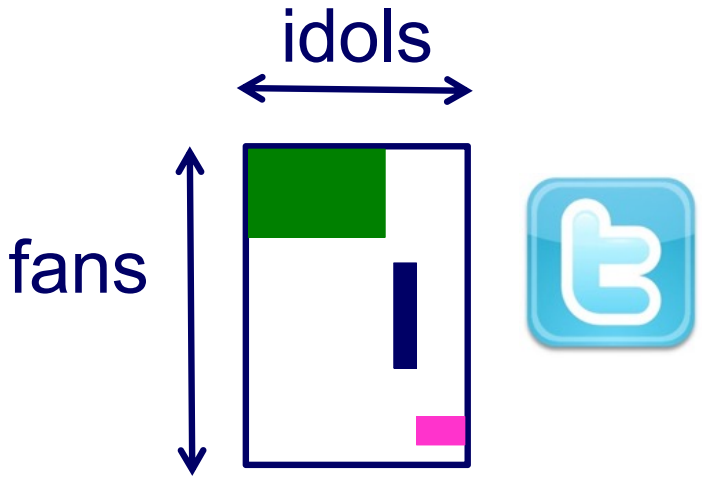
- Given a graph (eg., web pages containing the desirable query word)
- Q1: Which node is the most important?



Why not just PageRank?

1. HITS (and its derivative, SALSA), differentiate between "hubs" and "authorities"
2. HITS can help to find the largest community
3. (SVD: powerful tool)

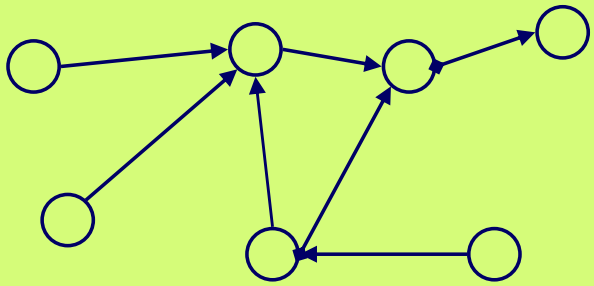




Problem: PageRank

From PR

Given a directed graph, find its most interesting/central node

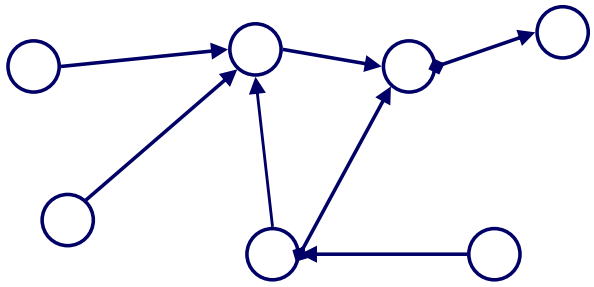


A node is important,
if its parents are important
(recursive, but OK!)

HITS

Problem: PageRank

Given a directed graph, find its most interesting/central node

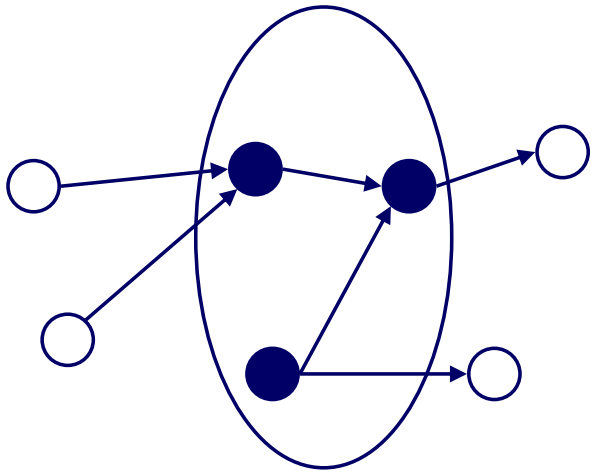


A node is important, ``wise''
if its parents are important
(recursive, but OK!)

AND: A node is ``wise''
if its children are important

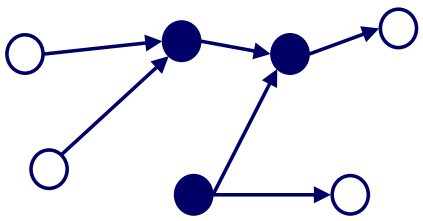
Kleinberg's algorithm

- Step 0: find nodes with query word(s)
- Step 1: expand by one move forward and backward

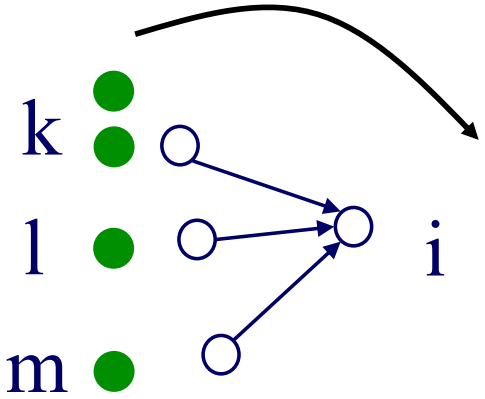


Kleinberg's algorithm

- on the resulting graph, give high score (= 'authorities') to nodes that many ``wise'' nodes point to
- give high wisdom score ('hubs') to nodes that point to good 'authorities'



Kleinberg's algorithm



Then:

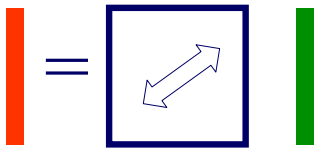
$$a_i = h_k + h_l + h_m$$

that is

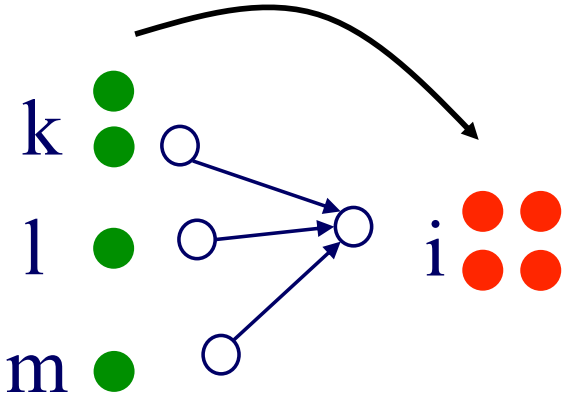
a_i = Sum (h_j) over all j that (j,i) edge exists

or

$$\mathbf{a} = \mathbf{A}^T \mathbf{h}$$



Kleinberg's algorithm



Then:

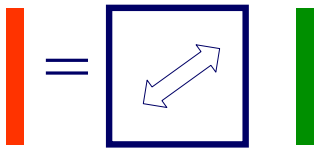
$$a_i = h_k + h_l + h_m$$

that is

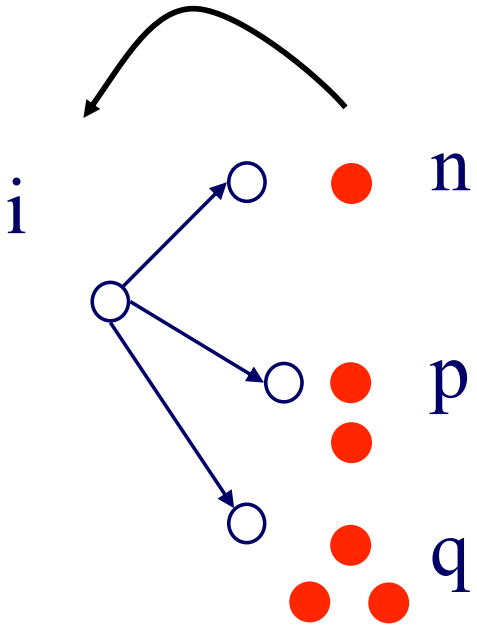
a_i = Sum (h_j) over all j that (j,i) edge exists

or

$$\mathbf{a} = \mathbf{A}^T \mathbf{h}$$



Kleinberg's algorithm



symmetrically, for the 'hubness':

$$h_i = a_n + a_p + a_q$$

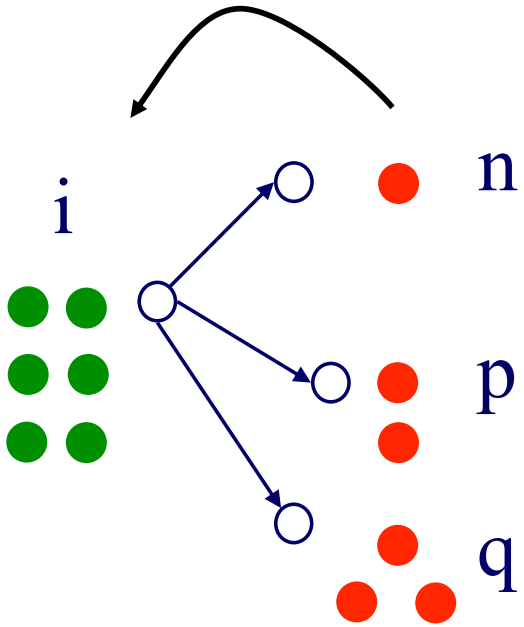
that is

h_i = Sum (q_j) over all j that (i,j) edge exists

or

$\mathbf{h} = \mathbf{A}\ \mathbf{a}$

Kleinberg's algorithm



symmetrically, for the 'hubness':

$$h_i = a_n + a_p + a_q$$

that is

h_i = Sum (q_j) over all j that (i,j) edge exists

or

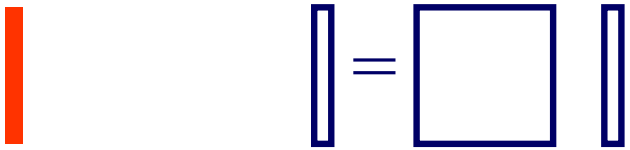
$$\mathbf{h} = \mathbf{A}\ \mathbf{a}$$

Kleinberg's algorithm

In conclusion, we want vectors **h** and **a** such that:

$$\mathbf{h} = \mathbf{A}\,\mathbf{a}$$

$$\mathbf{a} = \mathbf{A}^T\,\mathbf{h}$$



Kleinberg's algorithm

In conclusion, we want vectors **h** and **a** such that:

$$\mathbf{h} = \mathbf{A}\,\mathbf{a}$$

$$\mathbf{a} = \mathbf{A}^T\,\mathbf{h}$$

Kleinberg's algorithm

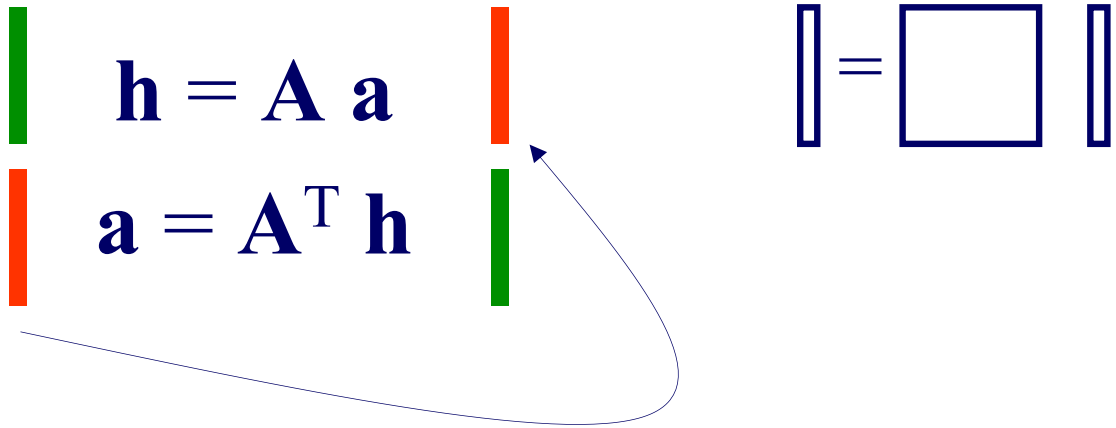
In conclusion, we want vectors **h** and **a** such that:

$$\mathbf{h} = \mathbf{A}\,\mathbf{a}$$

$$\mathbf{a} = \mathbf{A}^T\,\mathbf{h}$$

Kleinberg's algorithm

In conclusion, we want vectors **h** and **a** such that:



Kleinberg's algorithm

In short, the solutions to

$$\mathbf{h} = \mathbf{A}\,\mathbf{a}$$

$$\mathbf{a} = \mathbf{A}^T\,\mathbf{h}$$

Dfn: in +4

are the left- and right- singular-vectors of the adjacency matrix **A.**

Starting from random **a'** and iterating, we'll eventually converge

… to the vector of strongest singular value.

Kleinberg's algorithm - results

Eg., for the query 'java':

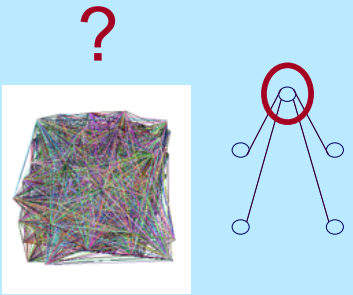
0.328 www.gamelan.com

0.251 java.sun.com

0.190 www.digitalfocus.com ("the java developer")

Bird's eye view

- Introduction – Motivation
- Part#1: (simple) Graphs
  - P1.1: node importance
    - PageRank and Personalized PR
    - HITS
    - SVD (Singular Value Decomposition)



SVD properties

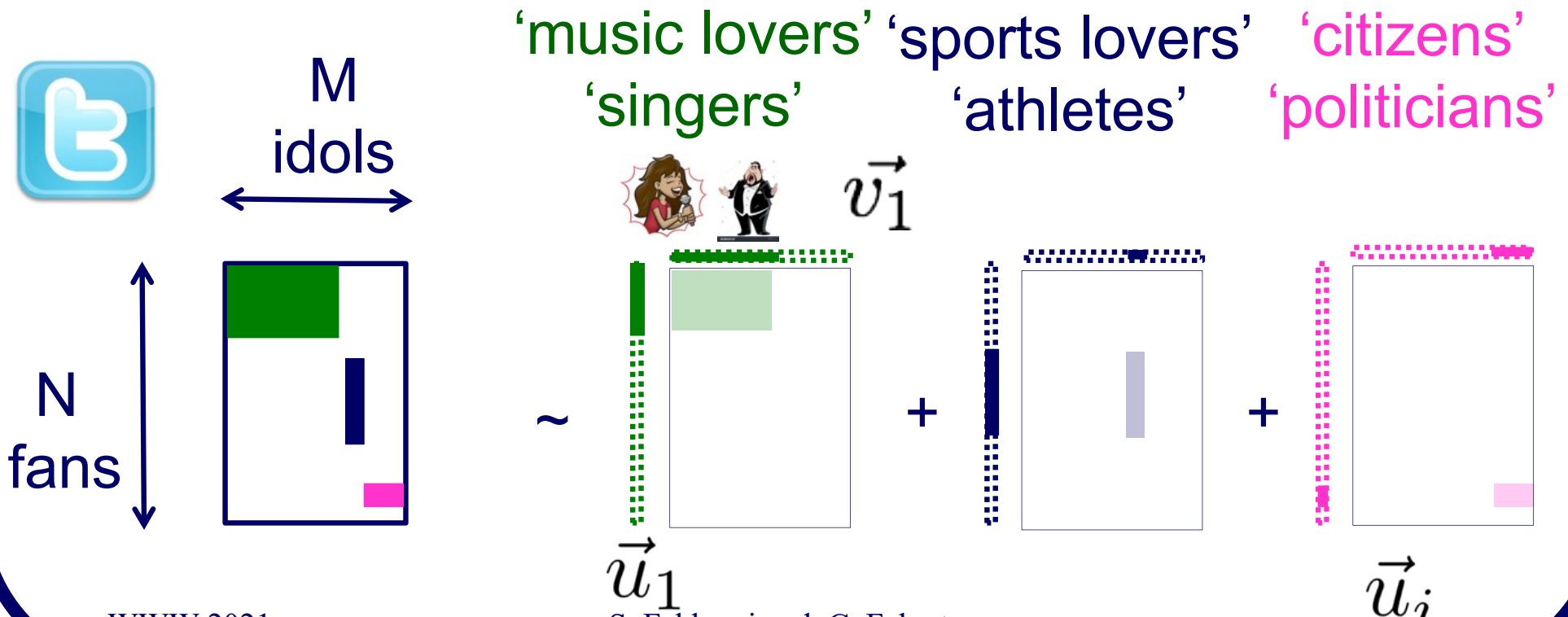
- Hidden/latent variable detection
- Compute node importance (HITS)
- Block detection
- Dimensionality reduction
- Embedding

$h = A\,a$
$a = A^T\,h$

Dfn: in +4

Crush intro to SVD

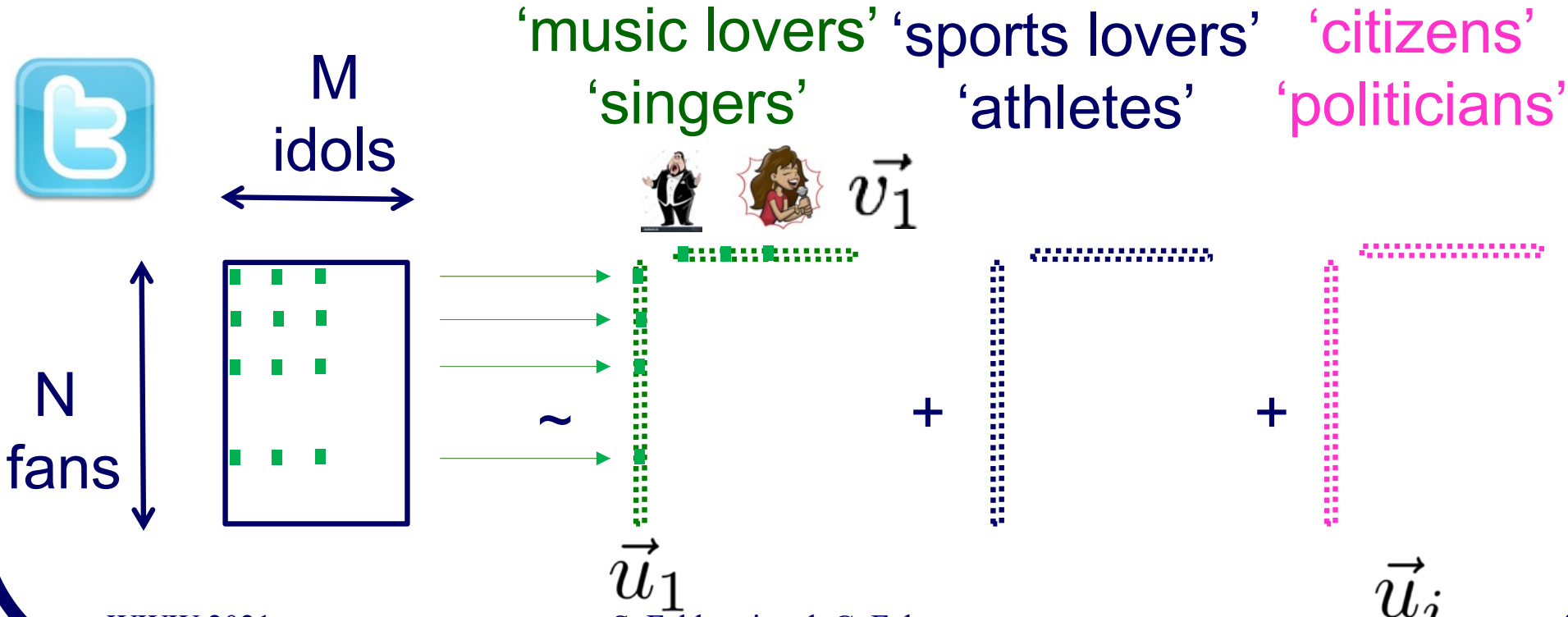
- (SVD) matrix factorization: finds blocks



Crush intro to SVD

- (SVD) matrix factorization: finds blocks

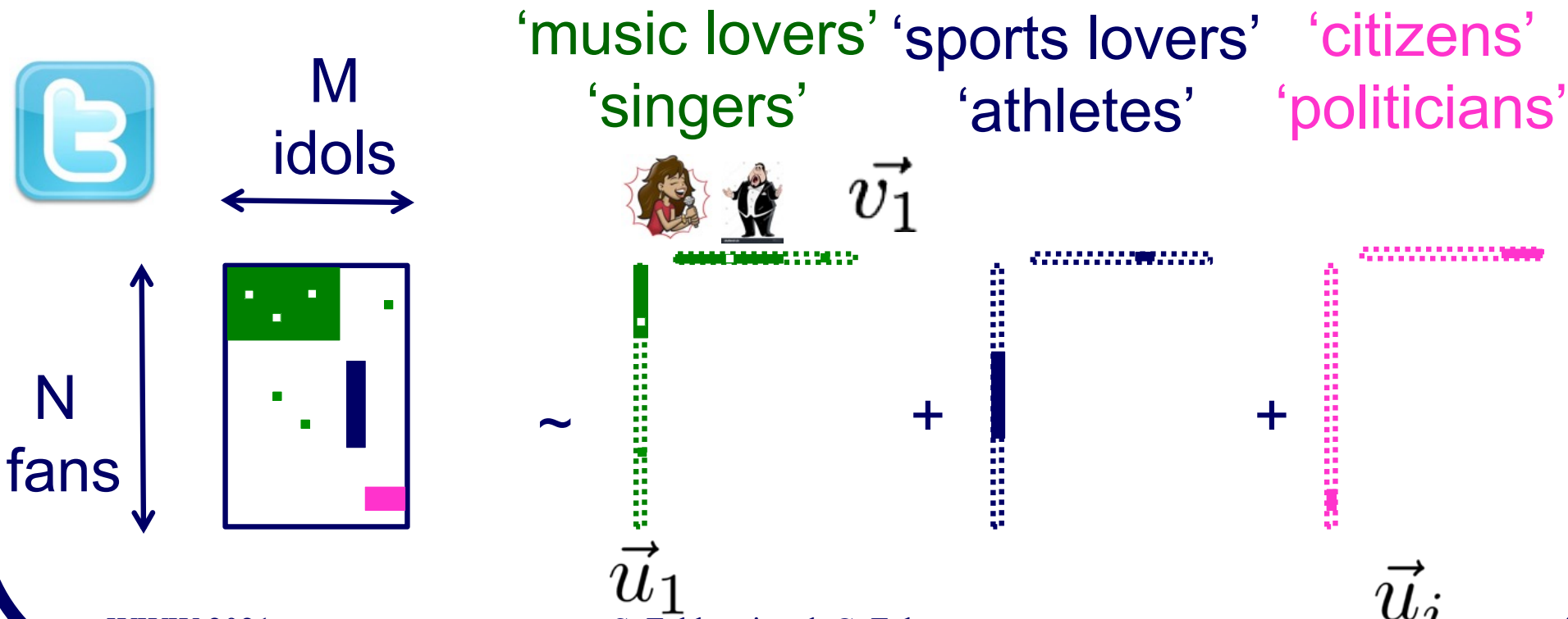
A) Even if shuffled!



Crush intro to SVD

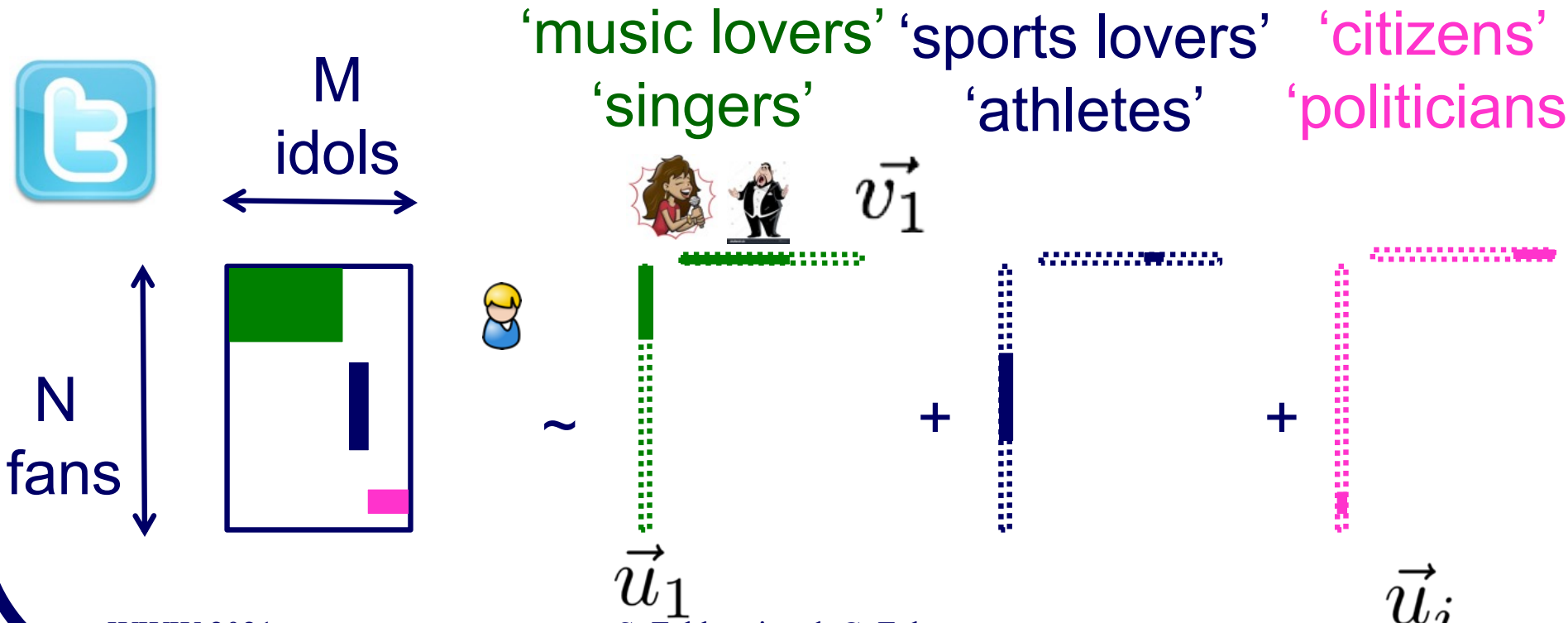
- (SVD) matrix factorization: finds blocks

B) Even if 'salt+pepper' noise



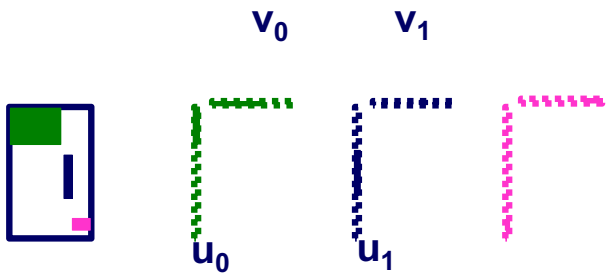
Crush intro to SVD

- Basis for anomaly detection – P1.3
- Basis for tensor/PARAFAC – P2.1



SVD properties

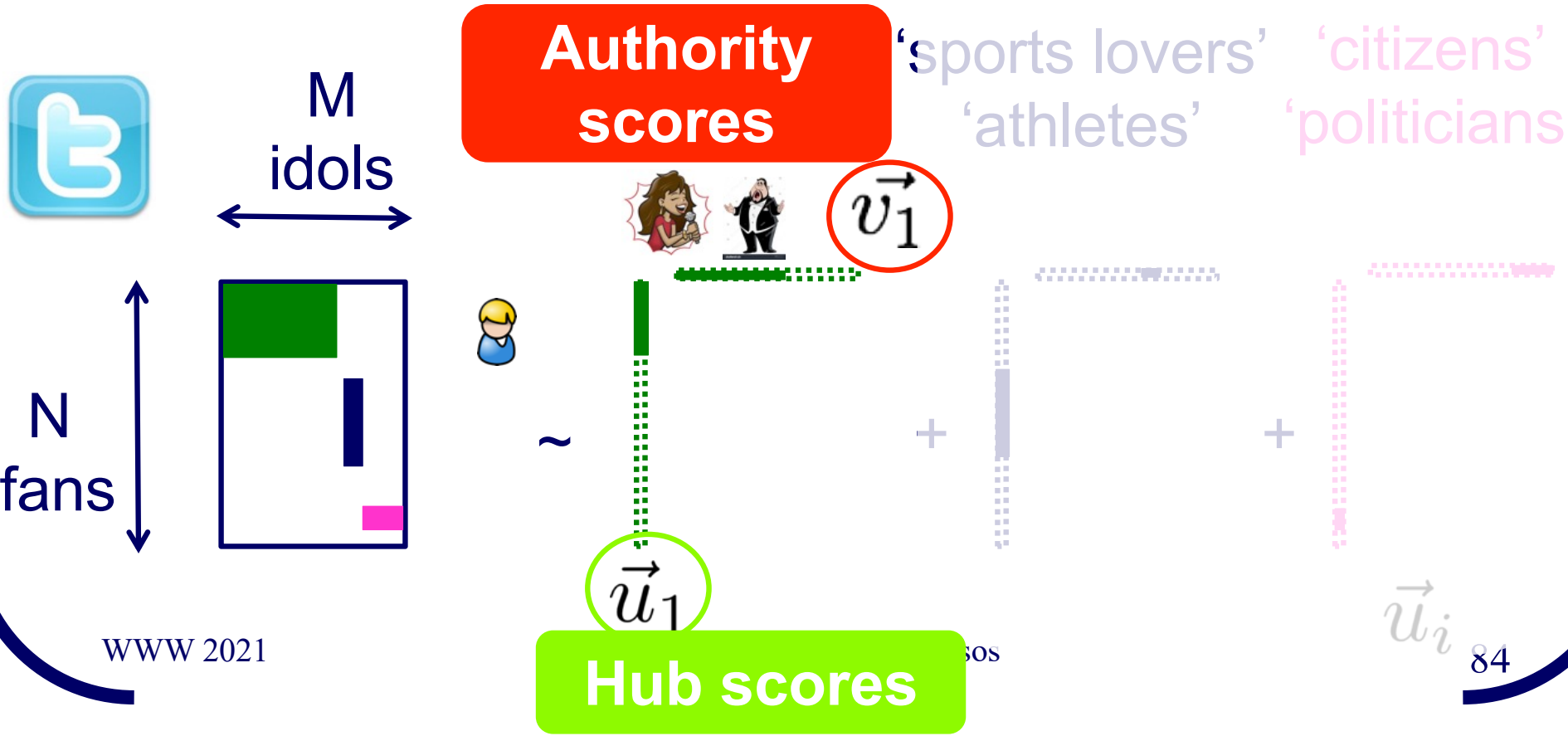
- ✓ Hidden/latent variable detection
- Compute node importance (HITS)
- Block detection
- Dimensionality reduction
- Embedding



Crush intro to SVD

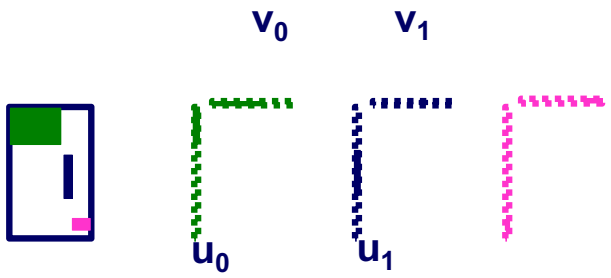
- (SVD) matrix factorization: finds blocks

HITS: first singular vector, ie, fixates on largest group



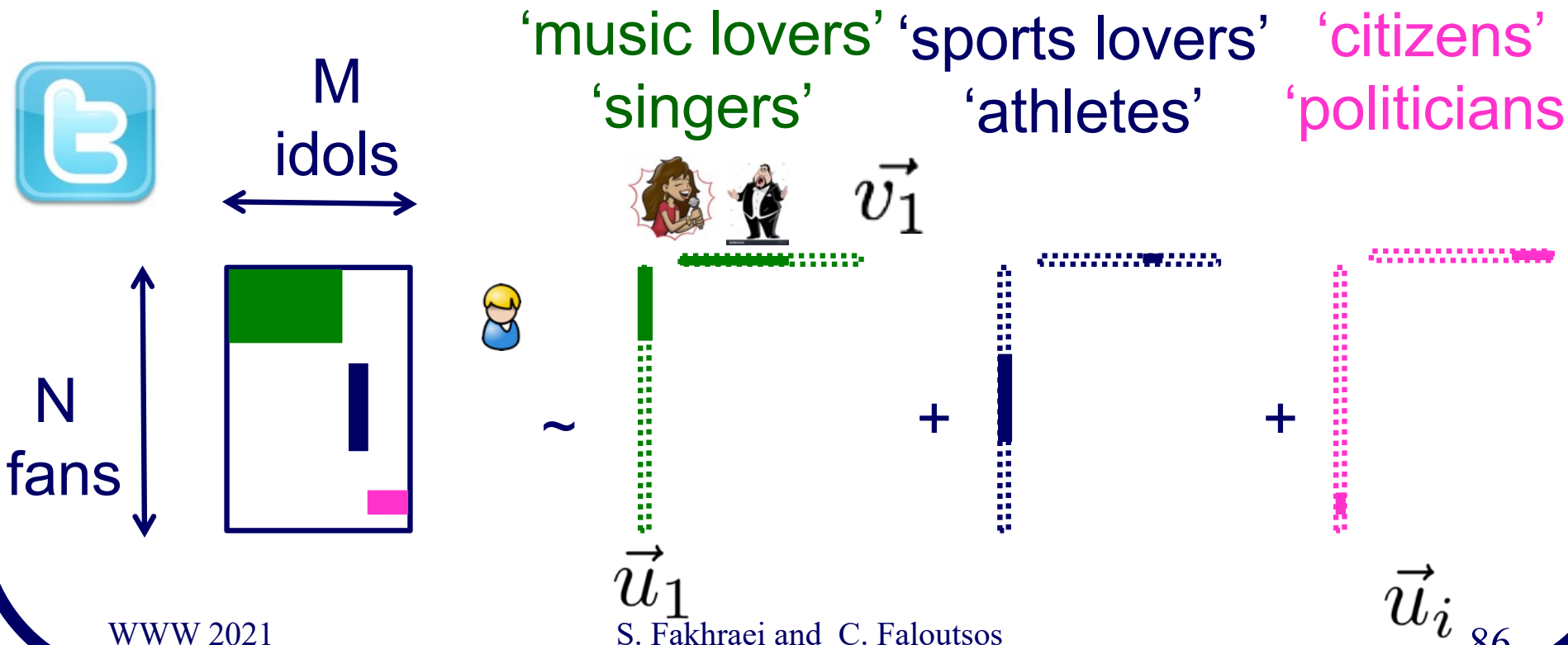
SVD properties

- ✓ Hidden/latent variable detection
- ✓ Compute node importance (HITS)
- Block detection
- Dimensionality reduction
- Embedding



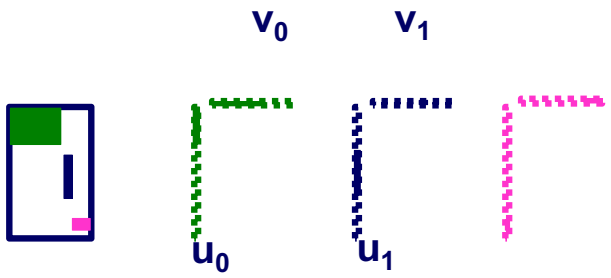
Crush intro to SVD

- (SVD) matrix factorization: finds blocks



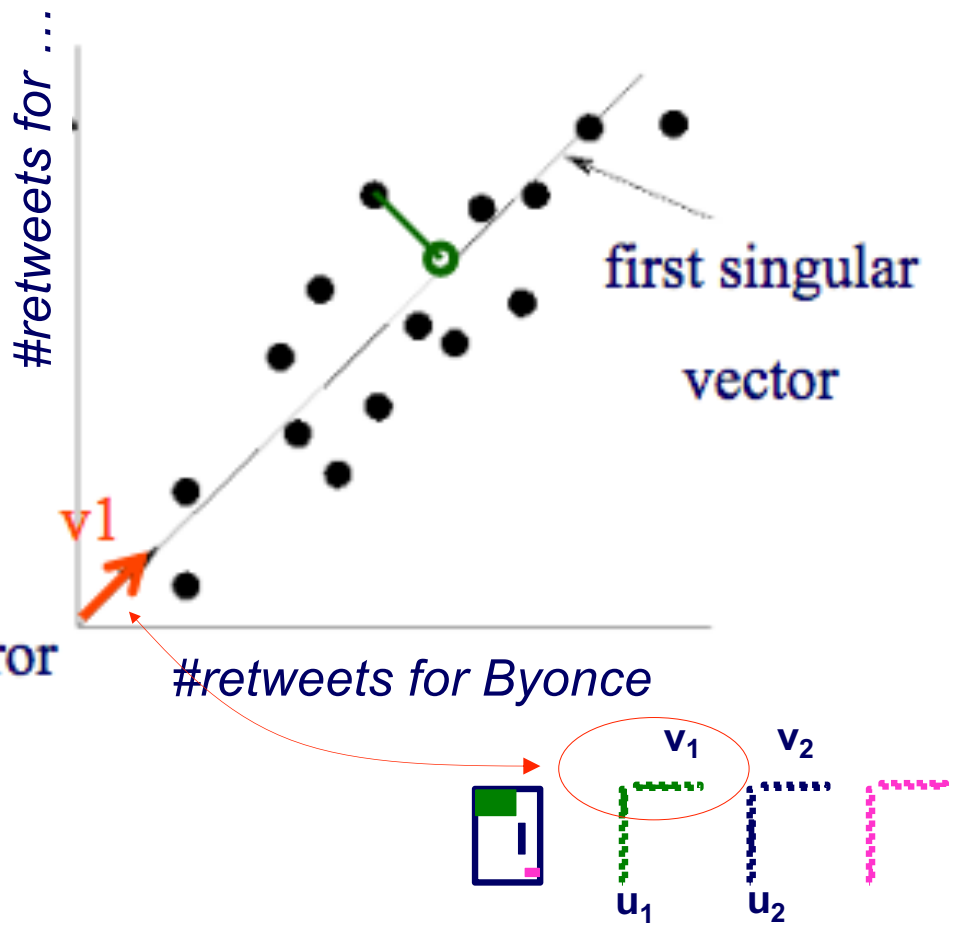
SVD properties

- ✓ Hidden/latent variable detection
- ✓ Compute node importance (HITS)
- ✓ Block detection
- Dimensionality reduction
- Embedding



SVD - intuition

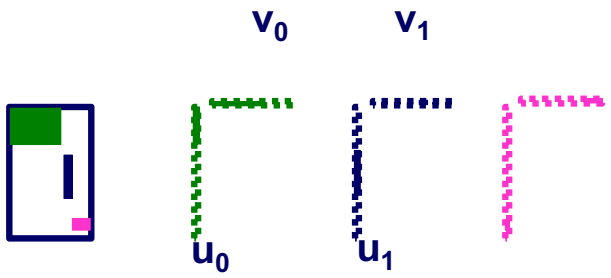
SVD: gives
best axis to project



- minimum RMS error

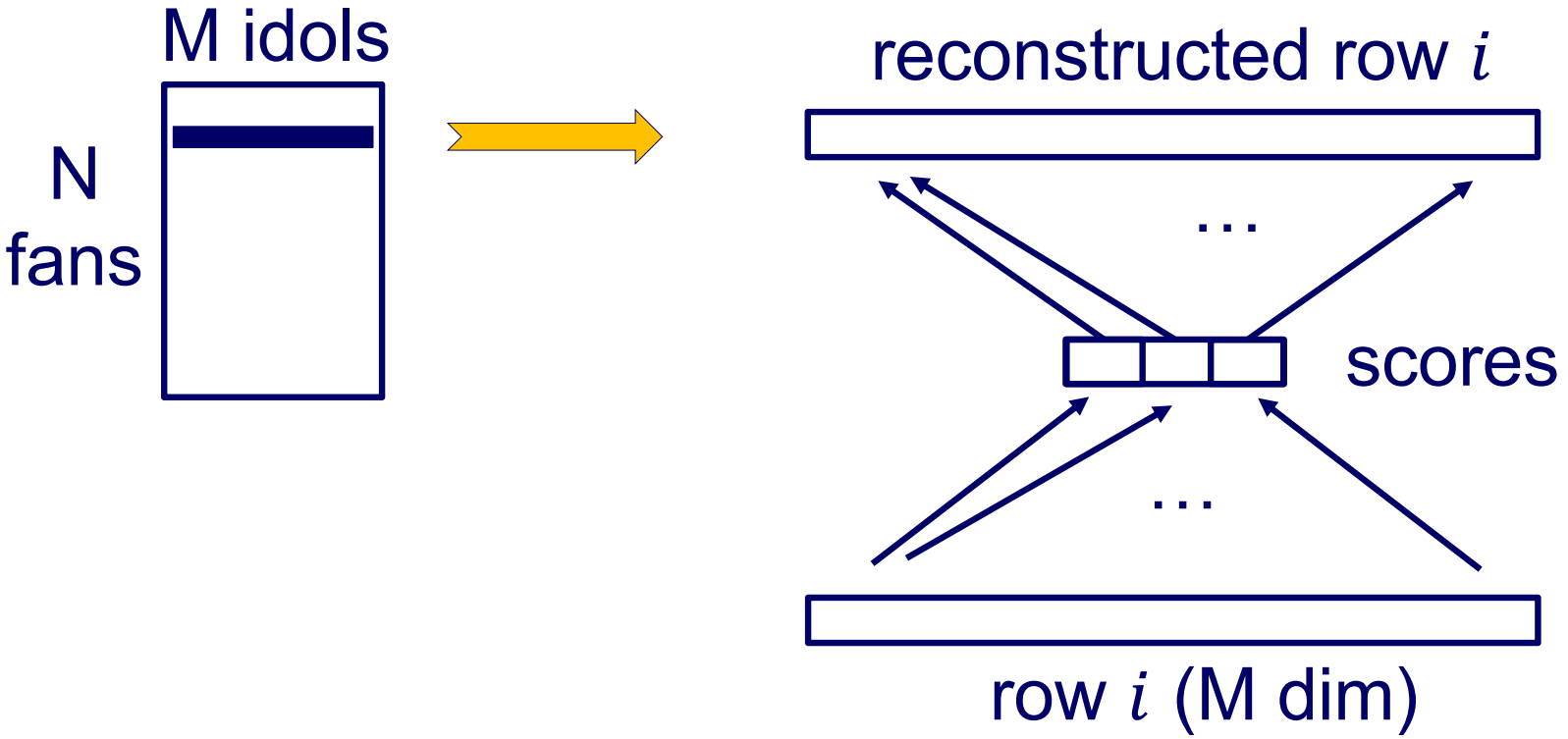
SVD properties

- ✓ Hidden/latent variable detection
- ✓ Compute node importance (HITS)
- ✓ Block detection
- ✓ Dimensionality reduction / projection
- Embedding



Crush intro to SVD

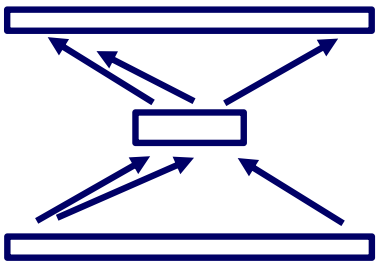
- SVD compression is a linear **autoencoder**

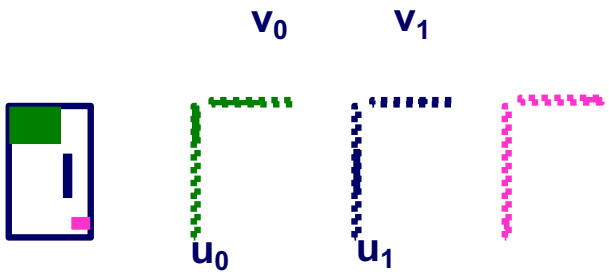


Independent Component Analysis, Aapo Hyvarinen, Erkki Oja, and Juha Karhunen (Wiley, 2001) – sec 6.2.4, p. 136.

SVD properties

- ✓Hidden/latent variable detection
- ✓Compute node importance (HITS)
- ✓Block detection
- ✓Dimensionality reduction
- ✓Embedding (linear)
  - SVD is a special case of 'deep neural net'





PoA talk on SVD

- https://broadcast.amazon.com/videos/141770

PoA Talk: Fraud Detection: Use the matrix Luke!

1994 ▶ 1191 ★★★★★

Presented by ehogue

Duration 1h 1min **Owned by** ehogue **Updated** 19 days ago **Recorded** about 1 year ago

Language English

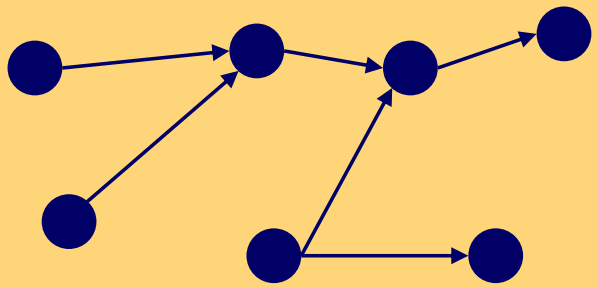
In Channel Principals of Amazon Talks

PoA Talk: Fraud Detection: Use the matrix Luke! - *Christos Faloutsos, Senior Principal Scientist, Transaction Risk Management*

How can we spot fraudulent Amazon customers, preemptively, without any prior, labeled set? Can we use the same tools, to spot fraudulent sellers? re-sellers? seller-buyer collusion? money laundering?

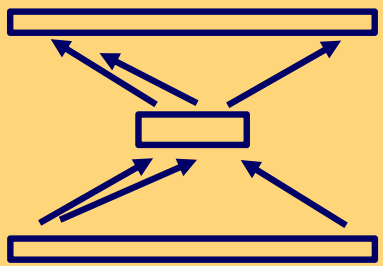
Node importance - Motivation:

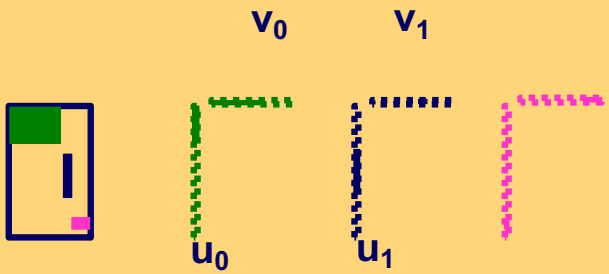
- Given a graph (eg., web pages containing the desirable query word)
- Q1: Which node is the most important?
  - **PageRank (PR = RWR)**, HITS
- Q2: How close is node 'A' to node 'B'?
  - **Personalized** P.R.



SVD properties

- ✓Hidden/latent variable detection
- ✓Compute node importance (HITS)
- ✓Block detection
- ✓Dimensionality reduction
- ✓Embedding (linear)
  - SVD is a special case of 'deep neural net'

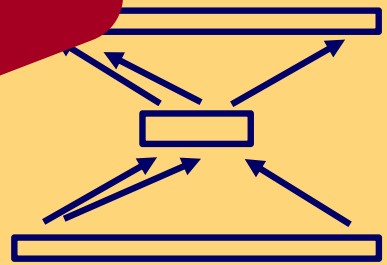


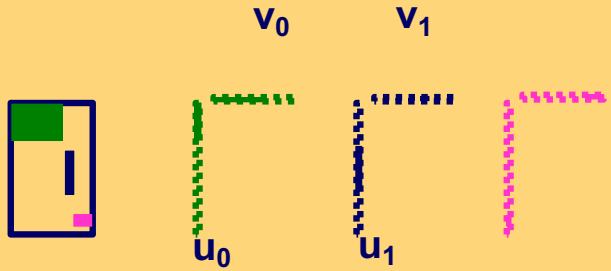


SVD properties

- ✓ Hidden/latent variable detection
- ✓ Compute node importance
- ✓ Block detection
- ✓ Dimensionality reduction
- ✓ Embedding (spectral)
  - SVD is a special case of 'deep neural net'

Matrix? SVD!





Bird's eye view

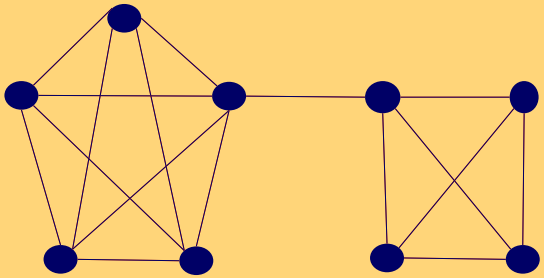
Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction		👍							
1.2 Comm. Detection									
1.3 Anomaly Detection									
1.4 Propagation									

Part 1: Plain Graphs

Part 2: Complex Graphs

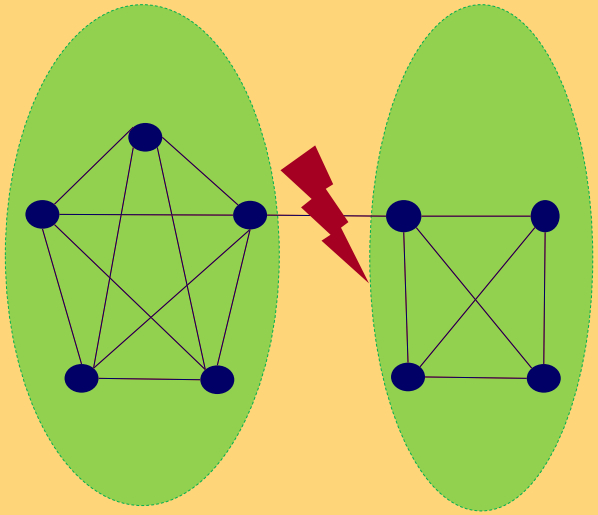
Problem

- Given a graph, and k
- Break it into k (disjoint) communities



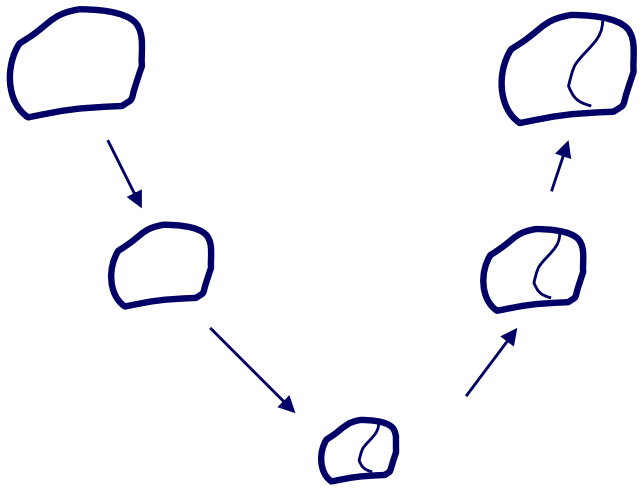
Short answer

- METIS [Karypis, Kumar]



Solution#1: METIS

- Arguably, the best algorithm
- Open source, at
  - http://glaros.dtc.umn.edu/gkhome/fetch/sw/metis/metis-5.1.0.tar.gz
- and *many* related papers, at same url
- Main idea:
  - coarsen the graph;
  - partition;
  - un-coarsen



Solution #1: METIS

- G. Karypis and V. Kumar. *METIS 4.0: Unstructured graph partitioning and sparse matrix ordering system*. TR, Dept. of CS, Univ. of Minnesota, 1998.
- <and many extensions>

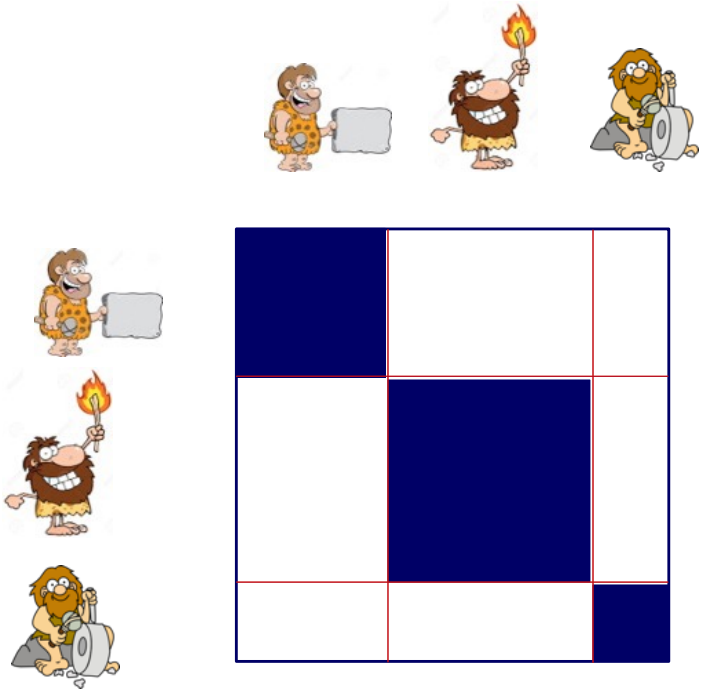
@ amazon aws

Solutions #2,3…

- **Fiedler vector** (2nd singular vector of Laplacian).
- **Modularity**: *Community structure in social and biological networks* M. Girvan and M. E. J. Newman, PNAS June 11, 2002. 99 (12) 7821-7826; https://doi.org/10.1073/pnas.122653799
- **Co-clustering**: [Dhillon+, KDD'03]
- **Clustering** on the $\mathbf{A}^2$ (square of adjacency matrix) [Zhou, Woodruff, PODS'04]
- **Minimum cut** / maximum flow [Flake+, KDD'00]
- ….

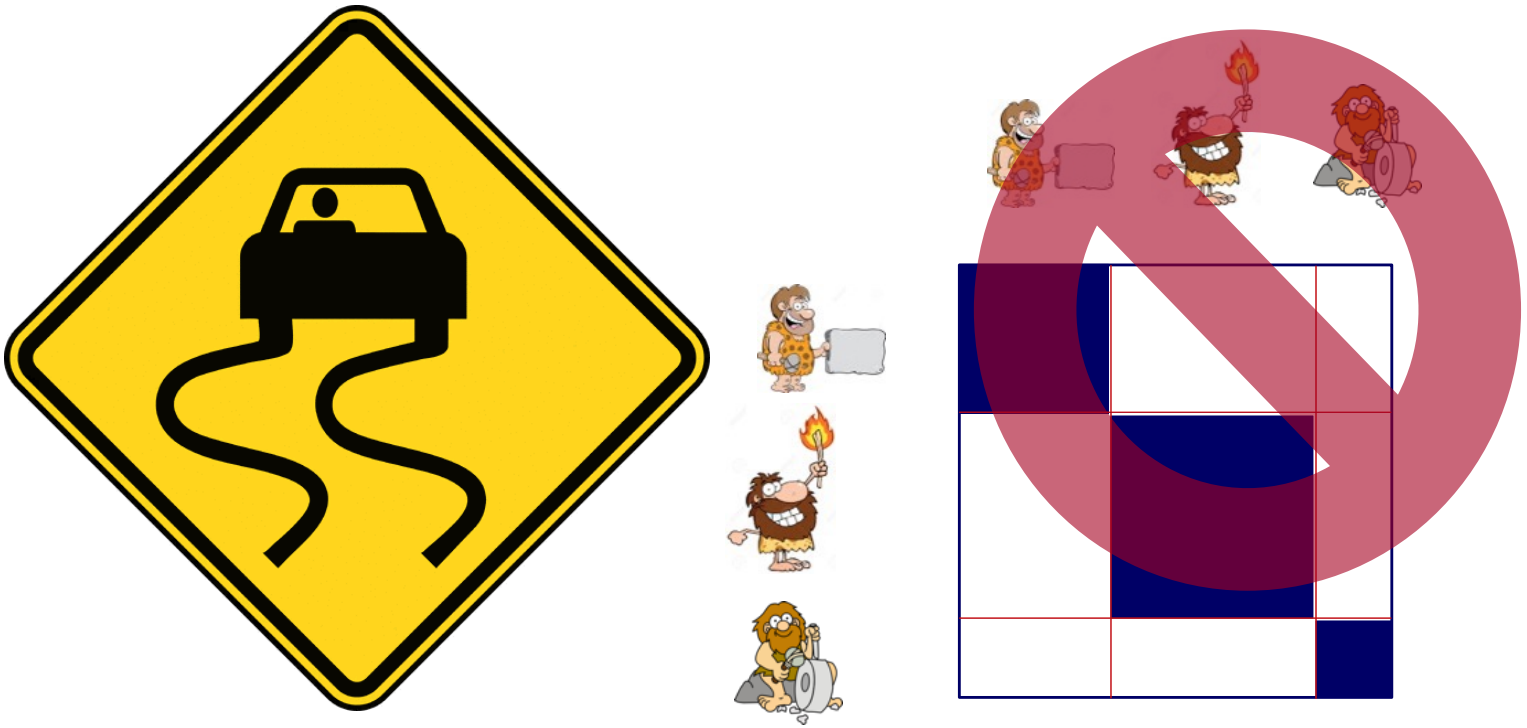
A word of caution

- BUT: often, there are **no good cuts**:



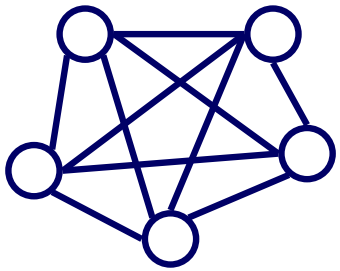
A word of caution

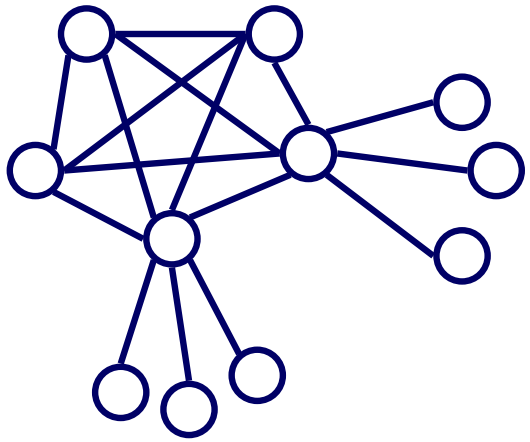
- BUT: often, there are **no good cuts**:



A word of caution

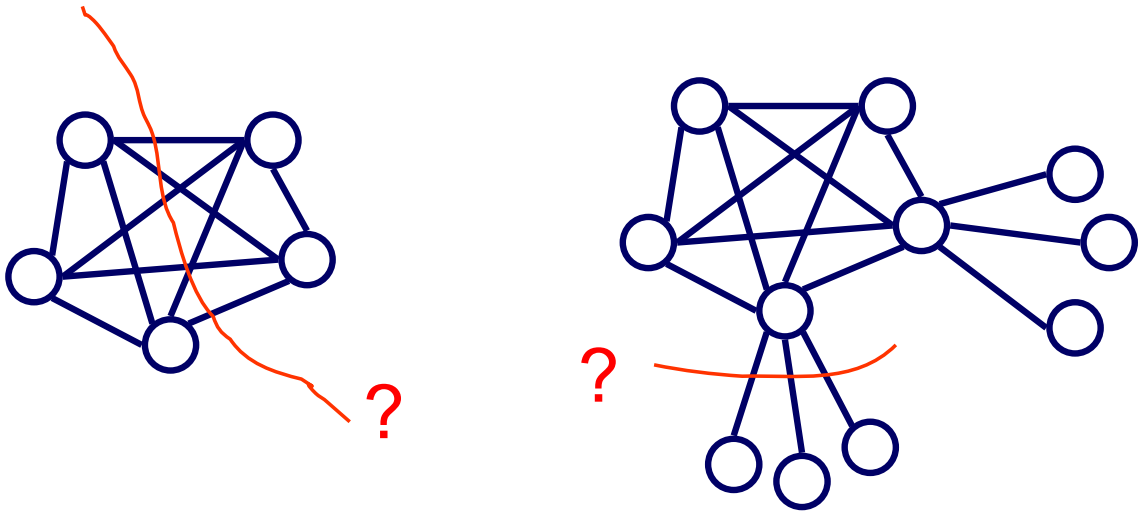
- Maybe there are no good cuts: ``jellyfish'' shape [Tauro+'01], [Siganos+,'06], strange behavior of cuts [Chakrabarti+'04], [Leskovec+,'08]





A word of caution

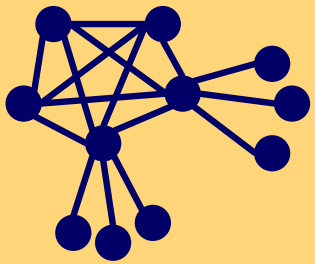
- Maybe there are no good cuts: ``jellyfish'' shape [Tauro+'01], [Siganos+,'06], strange behavior of cuts [Chakrabarti+,'04], [Leskovec+,'08]

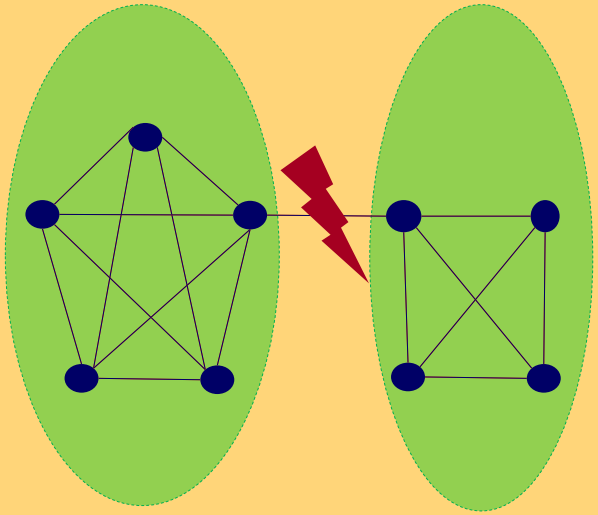


D. Chakrabarti, Y. Zhan, D. Blandford, C. Faloutsos and G. Blelloch: *NetMine: New Mining Tools for Large Graphs*, in SDM 2004 Workshop

Short answer

- METIS [Karypis, Kumar]
- (but: maybe NO good cuts exist!)





Bird's eye view

Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction		👍							
1.2 Comm. Detection			👍						
1.3 Anomaly Detection									
1.4 Propagation									

Part 1: Plain Graphs

Part 2: Complex Graphs

Bird's eye view

Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction		👍							
1.2 Comm. Detection			👍						
1.3 Anomaly Detection									
1.4 Propagation									

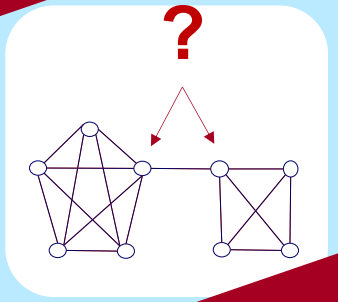
Part 1: Plain Graphs

Part 2: Complex Graphs

Bird's eye view

- Introduction – Motivation
- Part#1: (simple) Graphs
  - P1.1: node importance
  - P1.2: community detection
  - P1.3: fraud/anomaly detection
  - P1.4: belief propagation

un-supervised

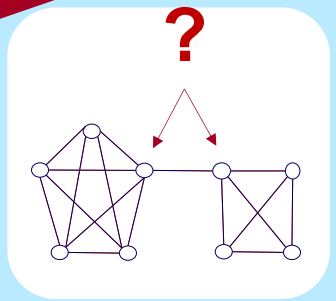


semi-supervised

Bird's eye view

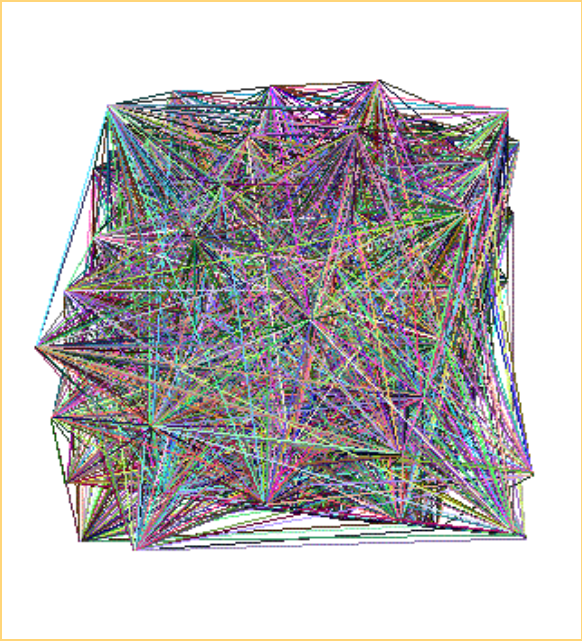
- Introduction – Motivation
- Part#1: (simple) Graphs
  - P1.1: node importance
  - P1.2: community detection
  - P1.3: fraud/anomaly detection
    - P1.3.1. Outliers
    - P1.3.2. Lock-step behavior
  - P1.4: belief propagation

un-supervised



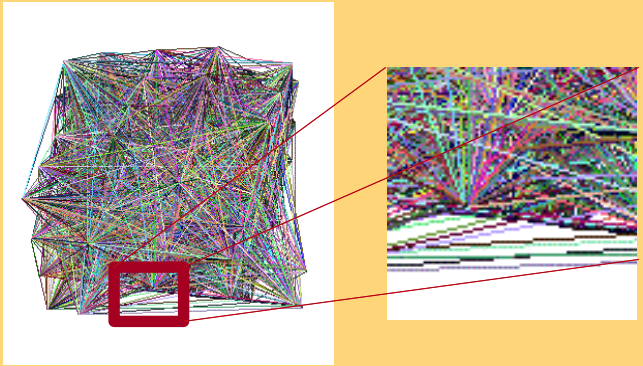
Problem

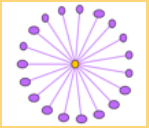
Given:

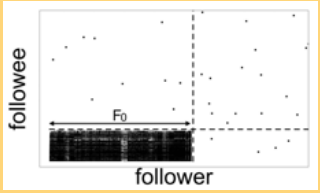


Find:

1) Outliers
2) Lock-step

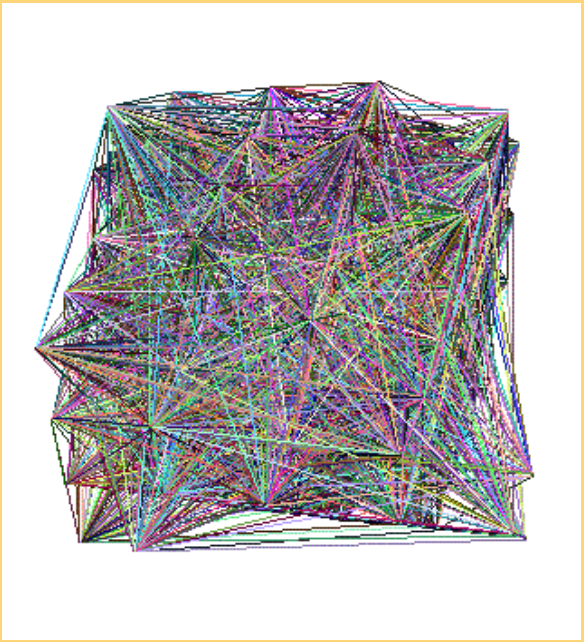






Solution

Given:

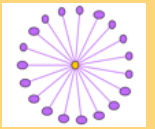


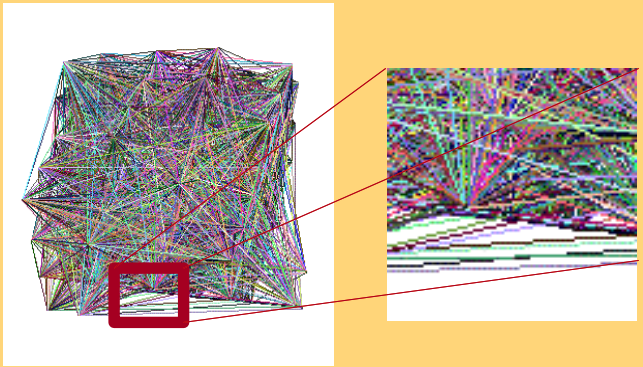
Find:

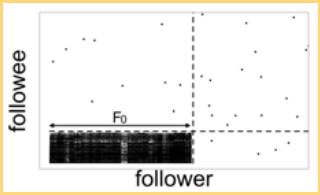
1) Outliers
2) Lock-step

OddBall

SVD

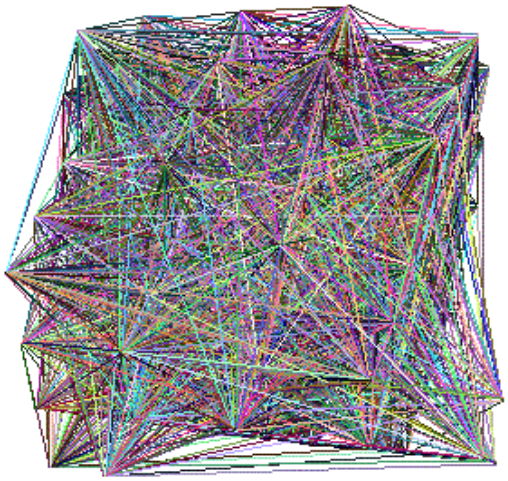






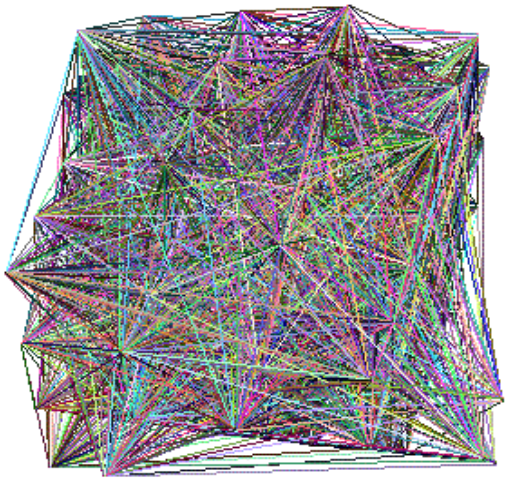
P1.3.1. Outliers

- Which node(s) are strange?
  - Q: How to start?

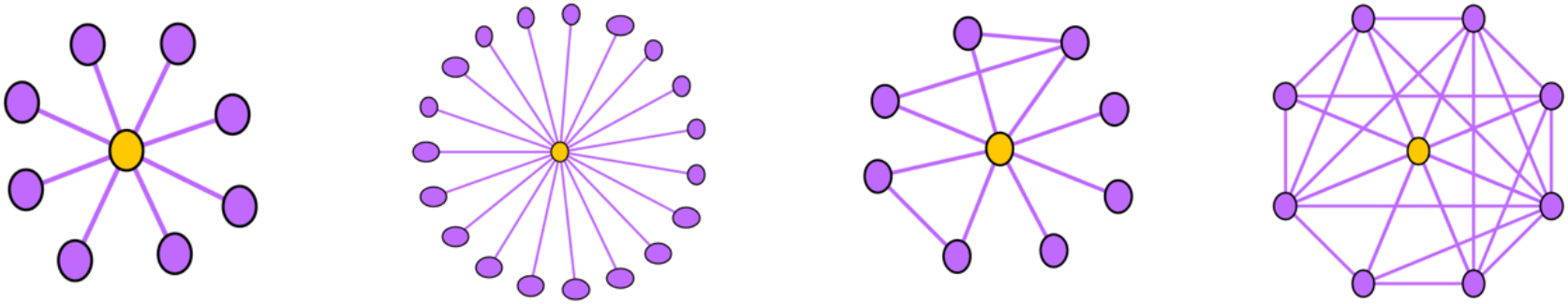


P1.3.1. Outliers

- Which node(s) are strange?
  - Q: How to start?
  - A1: egonet; and extract node features



Ego-net Patterns: Which is strange?

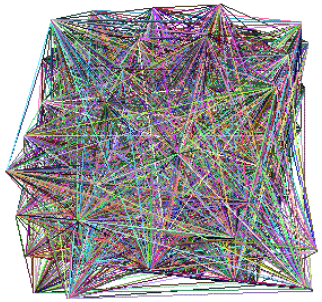




Oddball: Spotting anomalies in weighted graphs, Leman Akoglu, Mary McGlohon, Christos Faloutsos, *PAKDD* 2010

P1.3.1. Outliers

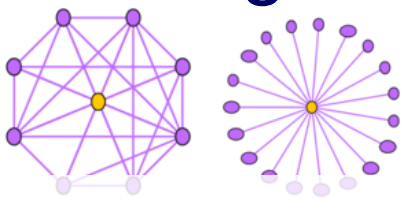
- Which node(s) are strange?
  - Q: How to start?
  - A: egonet; and extract node features
  - Q': which features?
  - A': ART! Infinite! Pick a few, e.g.:



KDD2020 ADS Panel: In ML
'*feature engineering is the hardest part*'

Ego-net Patterns

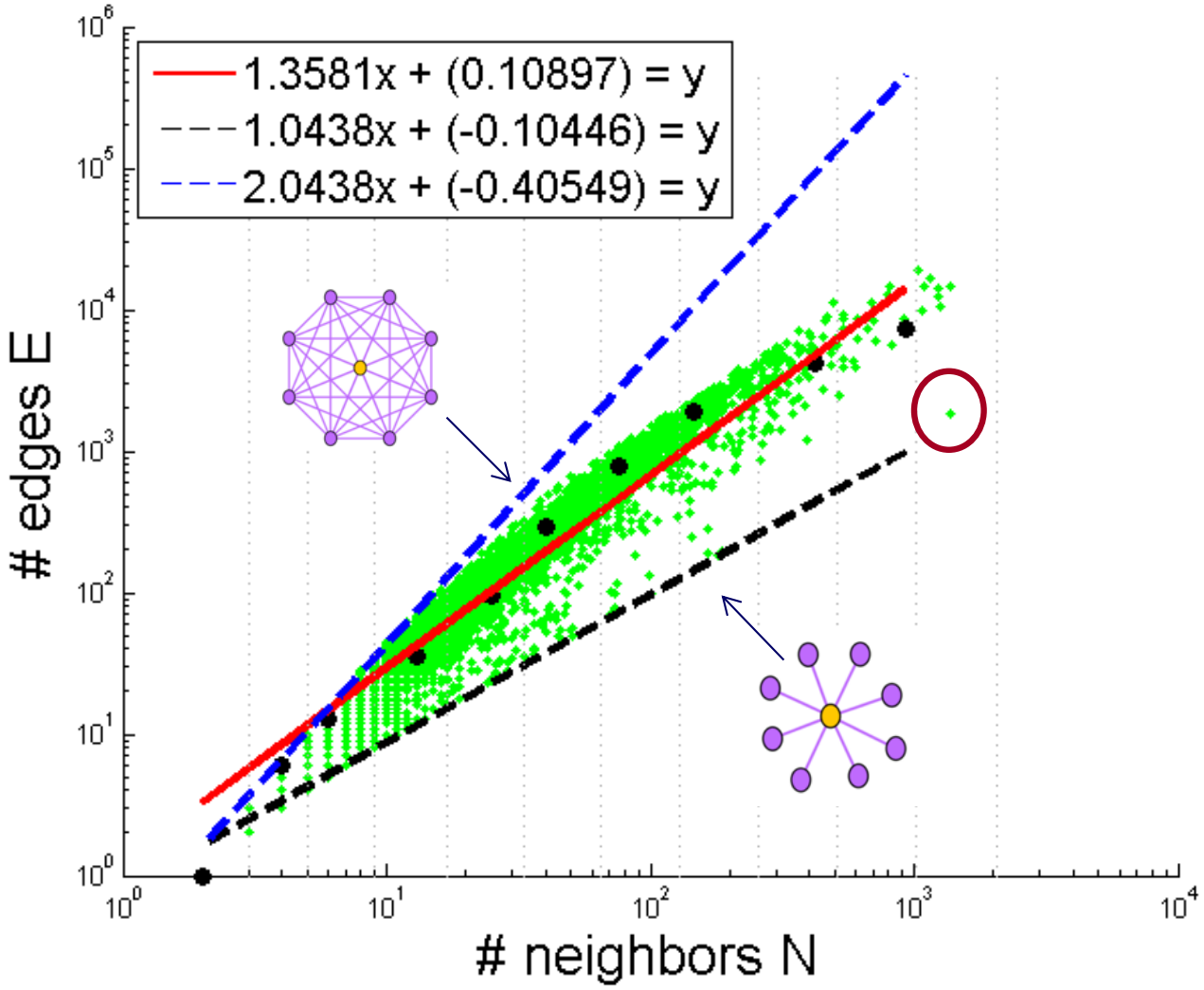
- N_i: number of neighbors (degree) of ego i
- E_i: number of edges in egonet i



- W_i: total weight of egonet i
- $\lambda_{w,i}$: principal eigenvalue of the weighted adjacency matrix of egonet i

Oddball: Spotting anomalies in weighted graphs, Leman Akoglu, Mary McGlohon, Christos Faloutsos, *PAKDD* 2010

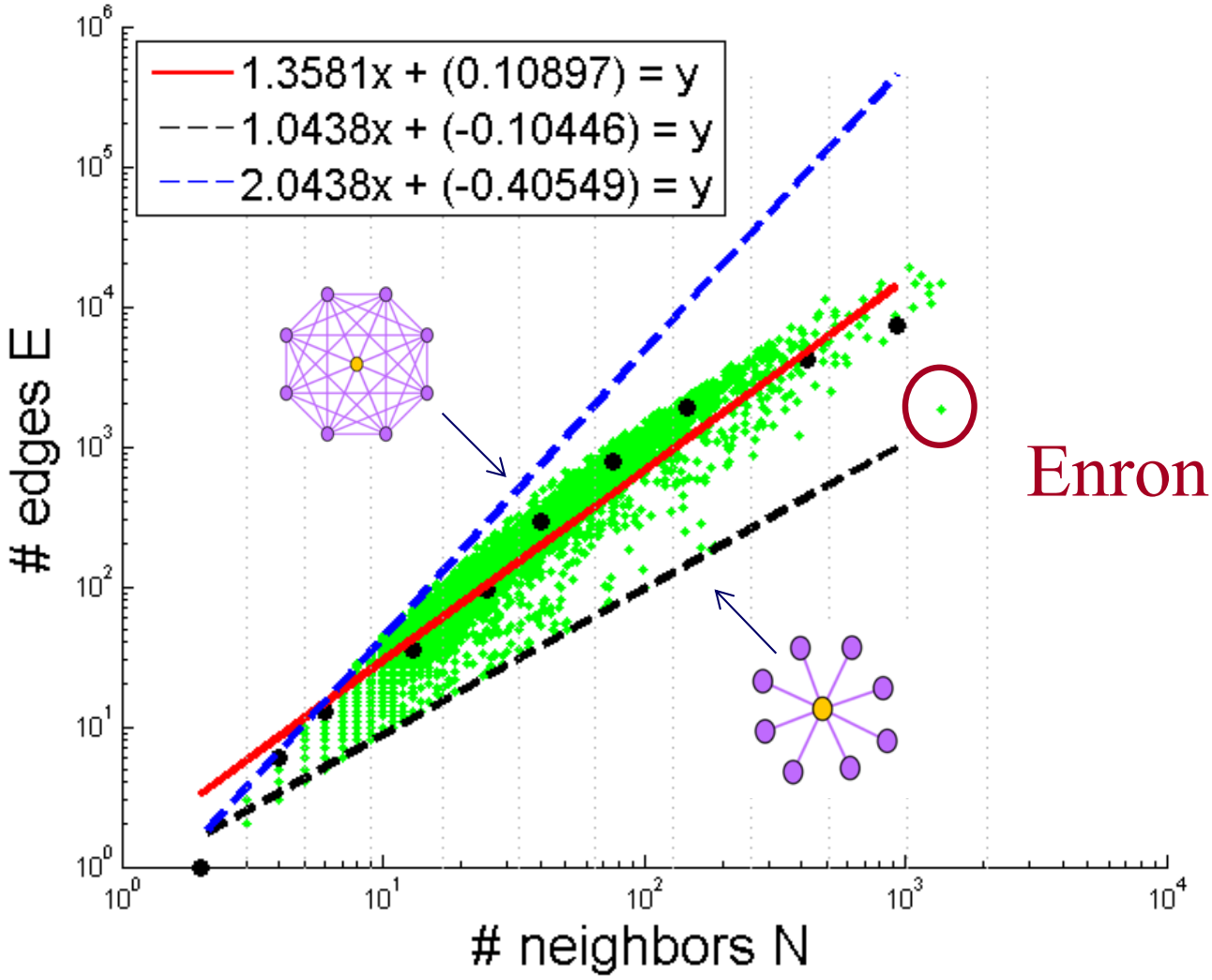
Pattern: Ego-net Power Law Density



$$E_i \propto N_i^{\alpha}$$
$$1 \leq \alpha \leq 2$$

Oddball: Spotting anomalies in weighted graphs, Leman Akoglu, Mary McGlohon, Christos Faloutsos, *PAKDD* 2010

Pattern: Ego-net Power Law Density



$$E_i \propto N_i^{\alpha}$$
$$1 \leq \alpha \leq 2$$

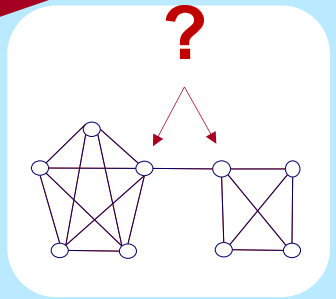
Enron CEO

Oddball: Spotting anomalies in weighted graphs, Leman Akoglu, Mary McGlohon, Christos Faloutsos, *PAKDD* 2010

Bird's eye view

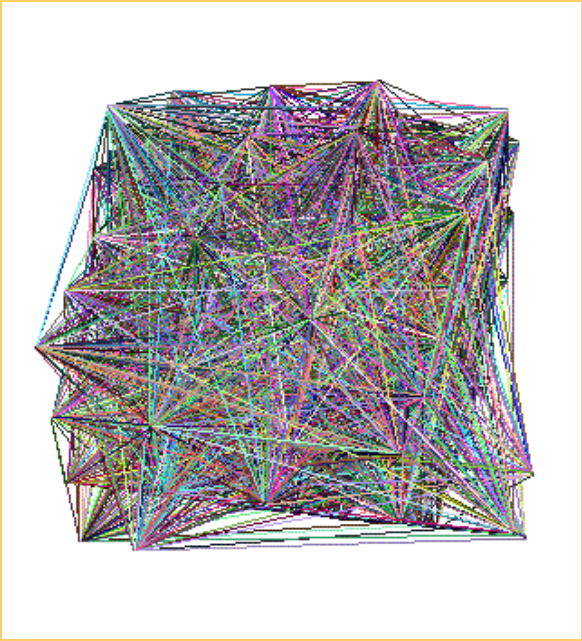
- Introduction – Motivation
- Part#1: (simple) Graphs
  - P1.1: node importance
  - P1.2: community detection
  - P1.3: fraud/anomaly detection
    - P1.3.1. Outliers
    - P1.3.2. Lock-step behavior
  - P1.4: belief propagation

un-supervised



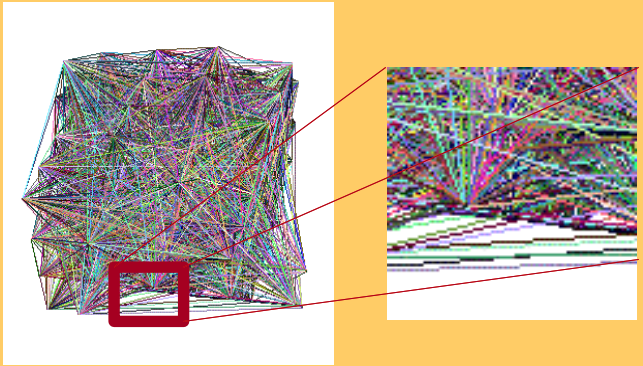
Problem

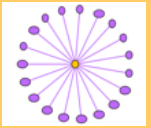
Given:

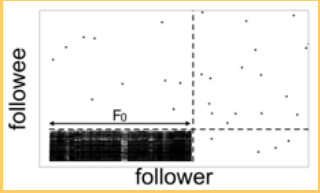


Find:

1) Outliers
2) Lock-step

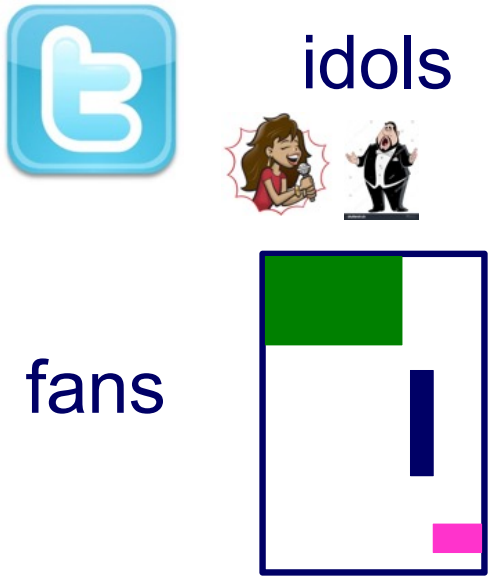






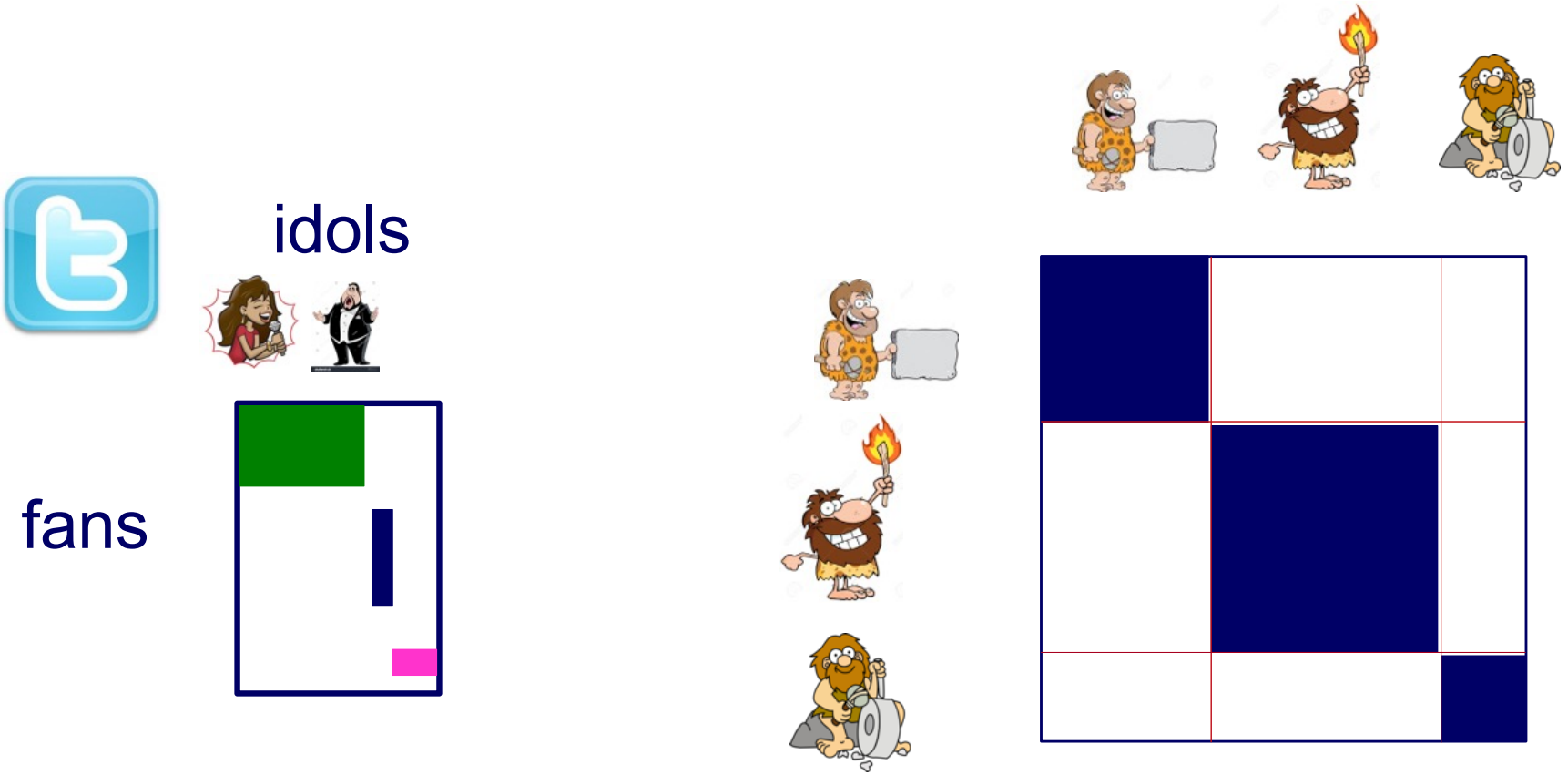
P1.3.1. How to find 'suspicious' groups?

- 'blocks' are normal, right?



P1.3.1. How to find 'suspicious' groups?

- 'blocks' are normal, right?

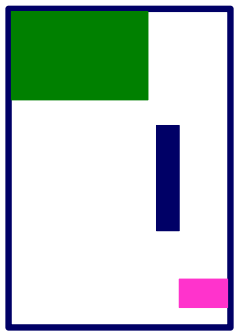


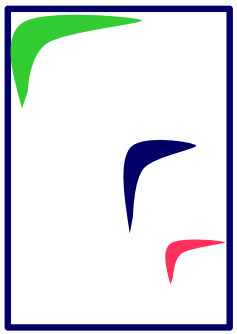
Except that:

- ‘blocks’ are normal, right?
- ‘hyperbolic’ communities are more realistic [Araujo+, PKDD’14]

idols

fans

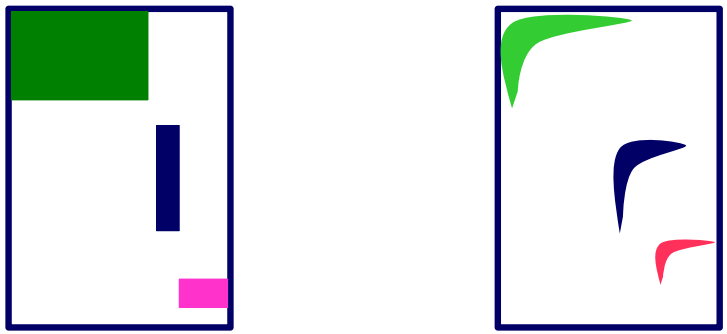




Except that:

- 'blocks' are usually suspicious
- 'hyperbolic' communities are more realistic [Araujo+, PKDD'14]

Q: Can we spot blocks, easily?

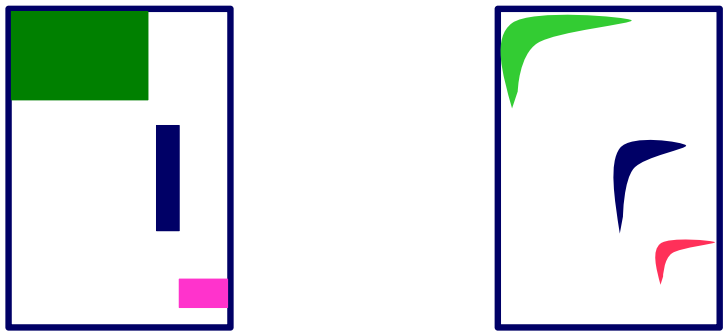


Except that:

- 'blocks' are usually suspicious
- 'hyperbolic' communities are more realistic [Araujo+, PKDD'14]

Q: Can we spot blocks, easily?

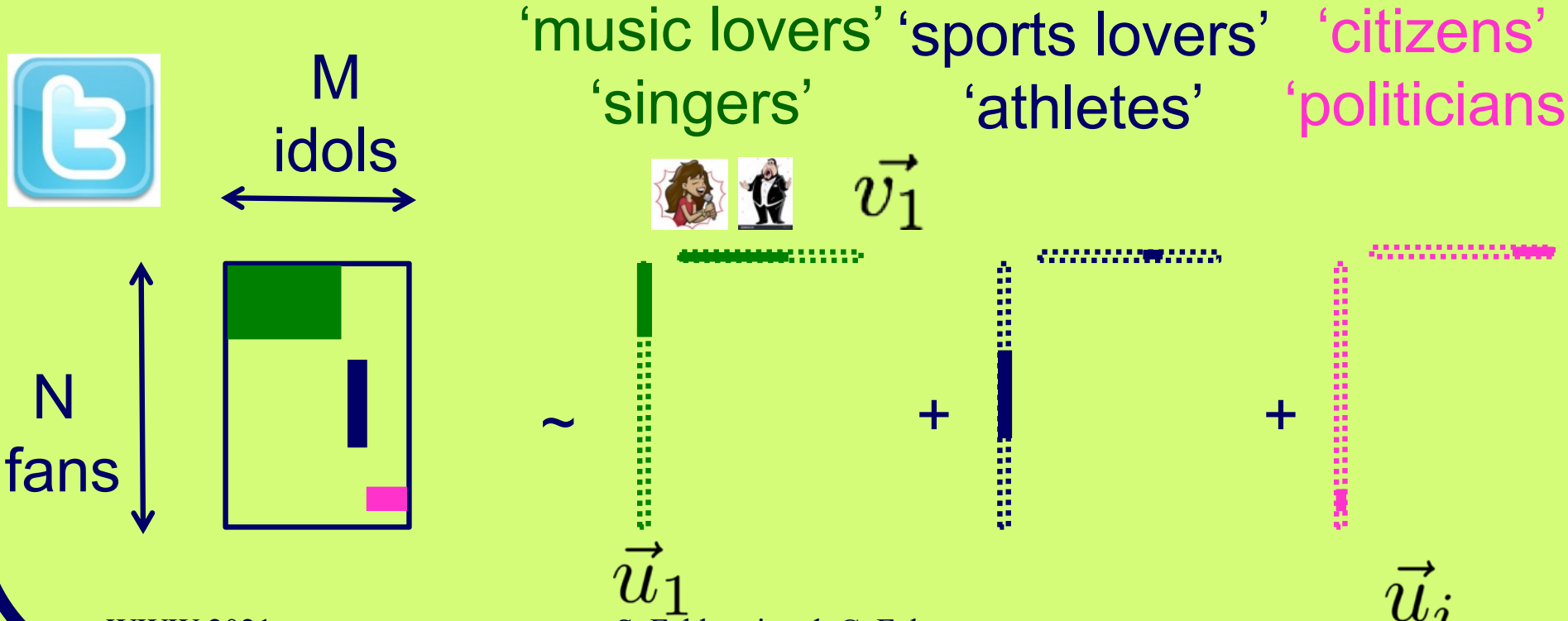
A: Silver bullet: SVD!



Crush intro to SVD

Reminder (from HITS)

- Recall: (SVD) matrix factorization: finds blocks



Inferring Strange Behavior from Connectivity Pattern in Social Networks

PAKDD'14



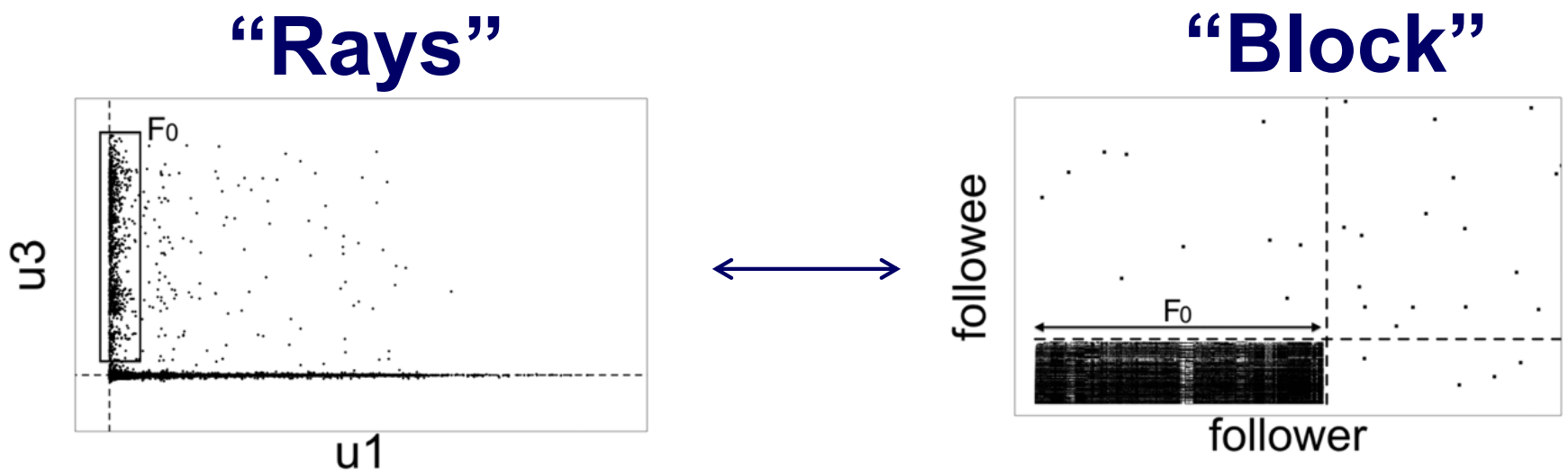
Meng Jiang, Peng Cui, Shiqiang Yang (Tsinghua)

Alex Beutel, Christos Faloutsos (CMU)

Dataset

- Tencent Weibo
- 117 million nodes (with profile and UGC data)
- 3.33 billion directed edges

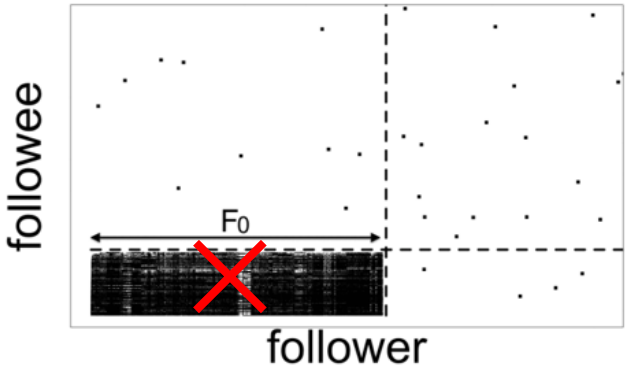
Real Data

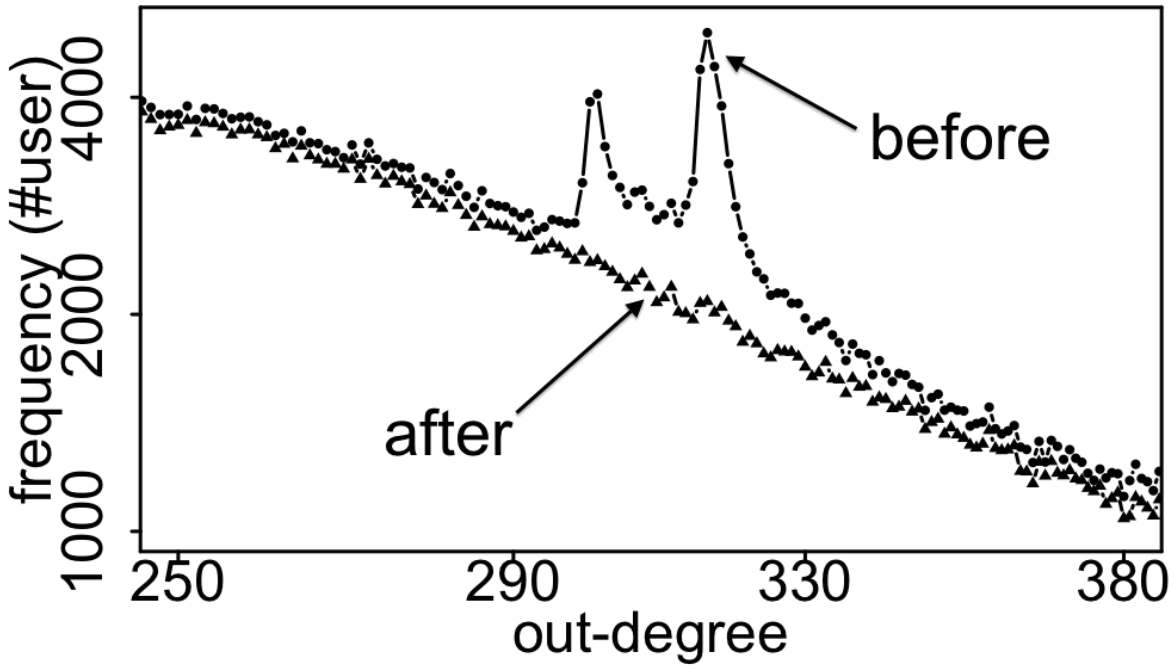


'blocks' create 'spokes'

Real Data

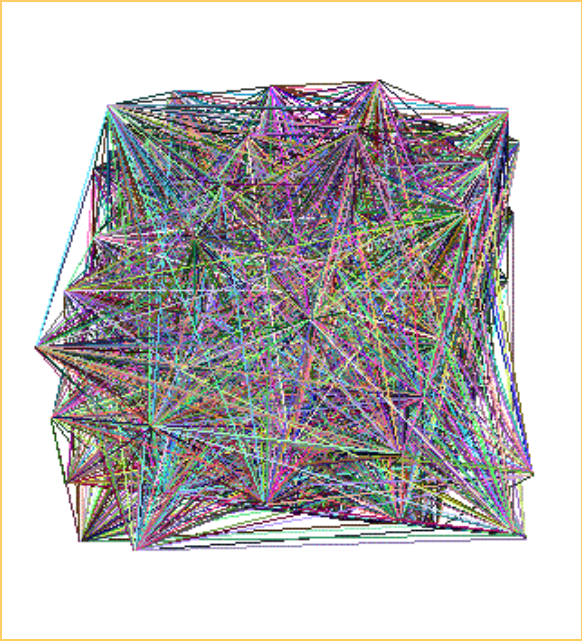
- Spikes on the out-degree distribution





Solution

Given:

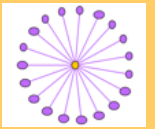


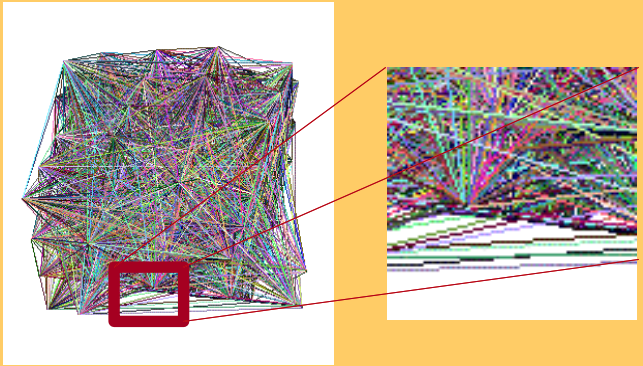
Find:

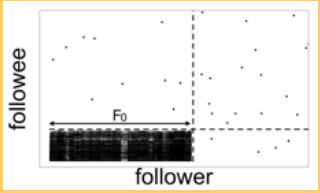
1) Outliers
2) Lock-step

OddBall

SVD







Bird's eye view

Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction		👍							
1.2 Comm. Detection			👍						
1.3 Anomaly Detection				👍					
1.4 Propagation									

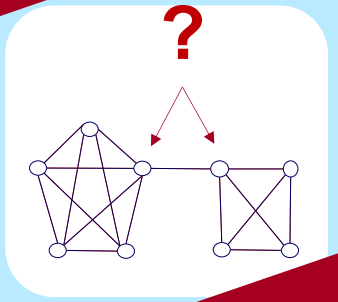
Part 1: Plain Graphs

Part 2: Complex Graphs

Bird's eye view

- Introduction – Motivation
- Part#1: (simple) Graphs
  - P1.1: node importance
  - P1.2: community detection
  - P1.3: fraud/anomaly detection
  - P1.4: belief propagation

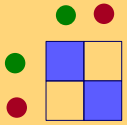
un-supervised

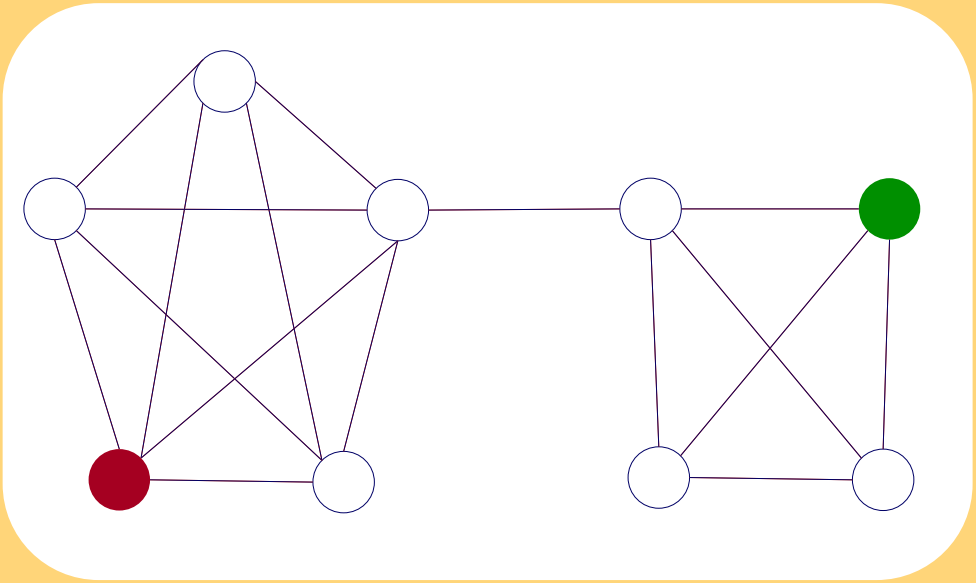


semi-supervised

Problem

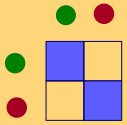
- What color, for the rest?
  - Given homophily (/heterophily etc)?

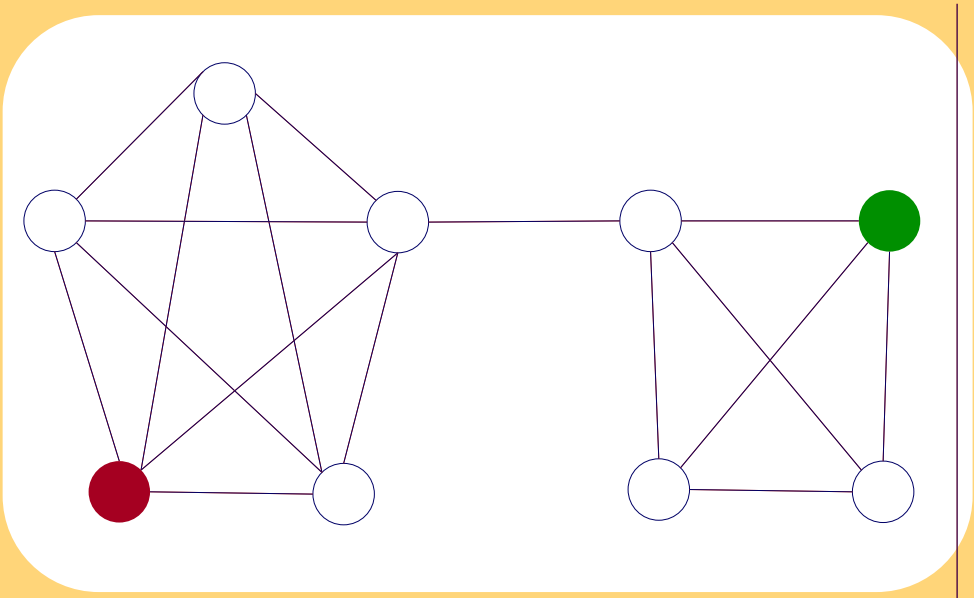




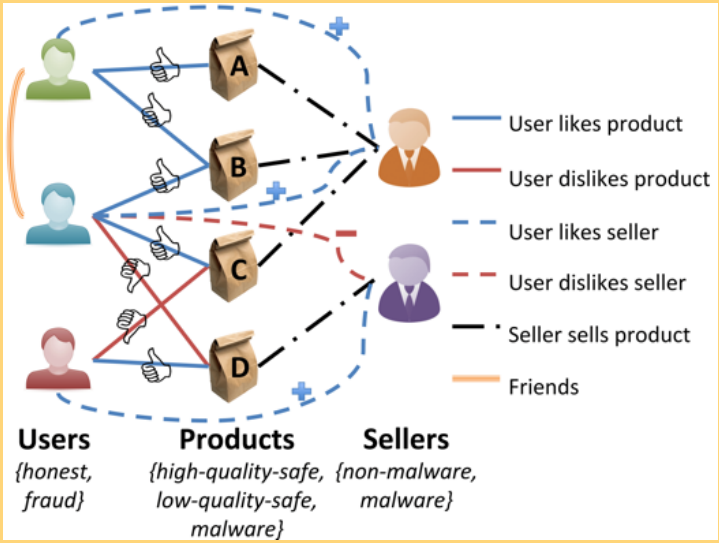
Short answer:

- What color, for the rest?
- A: Belief Propagation ('zooBP')



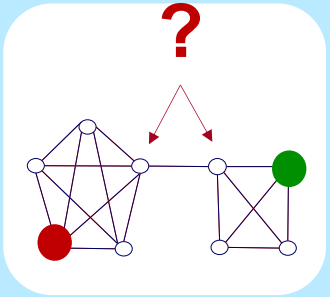


www.cs.cmu.edu/~deswaran/code/zoobp.zip



Bird's eye view

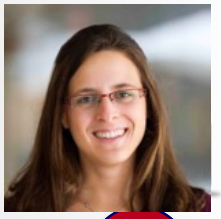
- Introduction – Motivation
- Part#1: (simple) Graphs
  - …
  - P1.4: belief propagation
    - Basics
    - Fast, linear approximation (FaBP)
    - Latest: zooBP
    - Success stories



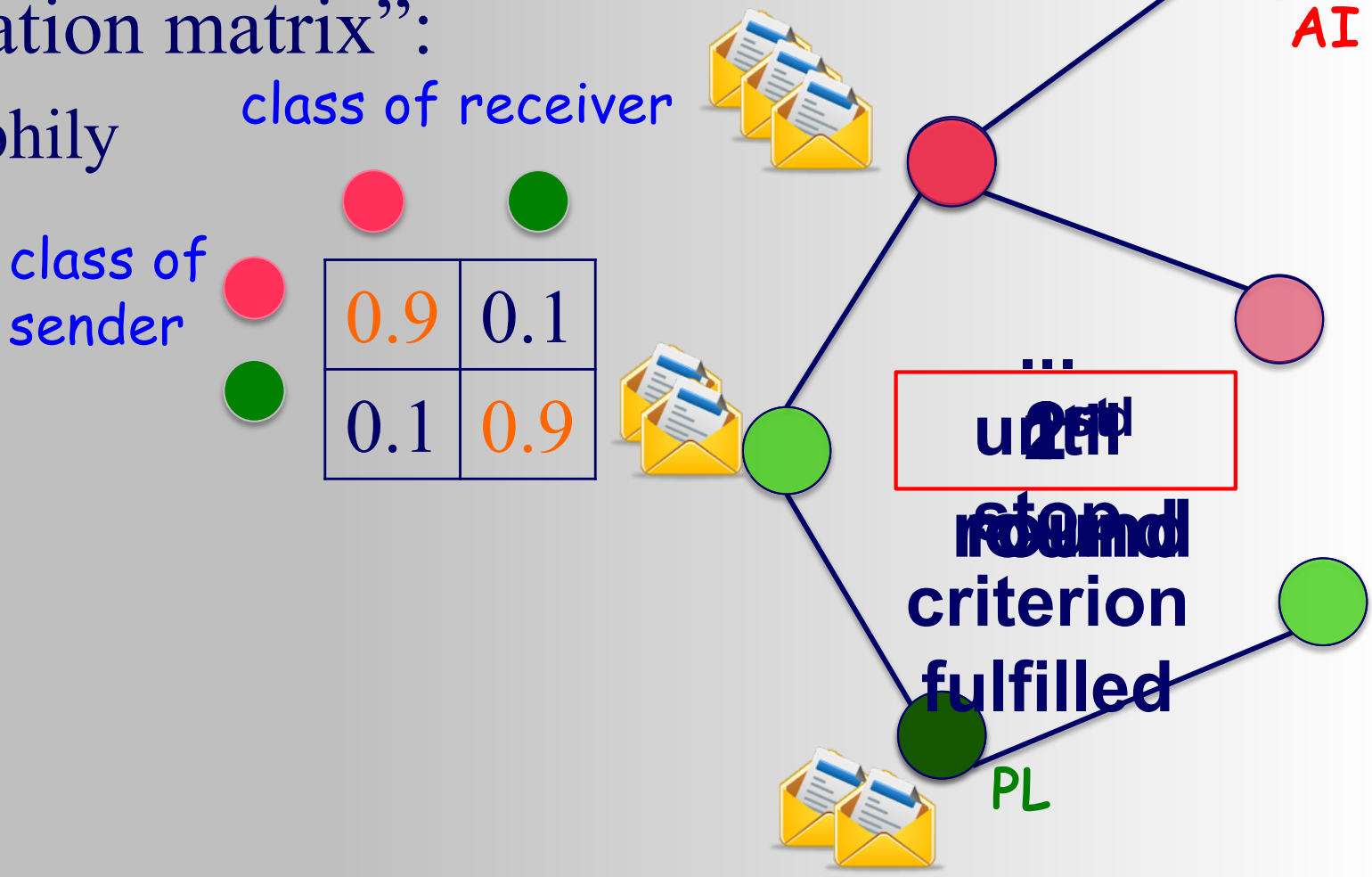
Prof. Danai Koutra
U. Michigan

Background

Belief Propagation



- Iterative message-based method
- "Propagation matrix":
  - Homophily

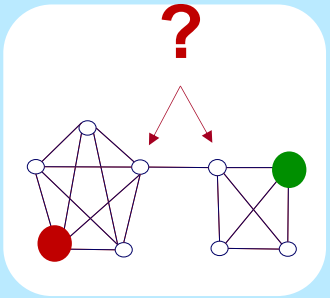


[Pearl '82][Yedidia+ '02] … [Gonzalez+ '09][Chechetka+ '10]

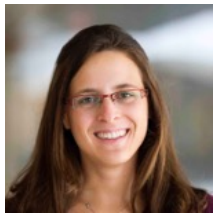
Background

Bird's eye view

- Introduction – Motivation
- Part#1: (simple) Graphs
  - …
  - P1.4: belief propagation
    - Basics
    - Fast, linear approximation (FaBP)
    - Latest: zooBP
    - Success stories

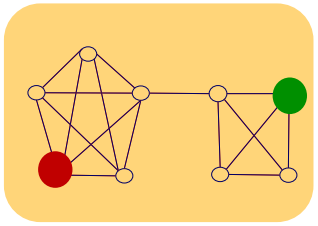


Unifying Guilt-by-Association Approaches: Theorems and Fast Algorithms



Danai Koutra
U Kang
Hsing-Kuo Kenneth Pao
Tai-You Ke
Duen Horng (Polo) Chau
Christos Faloutsos

ECML PKDD, 5-9 September 2011, Athens, Greece



BP vs. Linearized BP

DETAILS

Original [Yedidia+]:

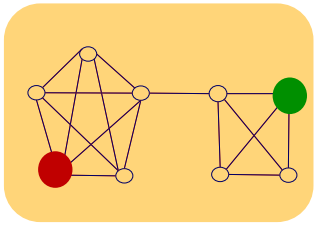
Belief Propagation

$$m_{ij}(x_j) \leftarrow \sum_{x_i} \phi_i(x_i) \cdot \psi_{ij}(x_i, x_j) \cdot \prod_{n \in N(i) \setminus j} m_{ni}(x_i)$$

$$b_i(x_i) \leftarrow \cdot \phi_i(x_i) \cdot \prod_{j \in N(i)} m_{ij}(x_i)$$

non-linear

- Closed-form formula?
- Convergence?



BP vs. Linearized BP

Original [Yedidia+]:

Belief Propagation

$$m_{ij}(x_j) \leftarrow \sum_{x_i} \phi_i(x_i) \cdot \psi_{ij}(x_i, x_j) \cdot \prod_{n \in N(i) \setminus j} m_{ni}(x_i)$$

$$b_i(x_i) \leftarrow \cdot \phi_i(x_i) \cdot \prod_{j \in N(i)} m_{ij}(x_i)$$

non-linear

Our proposal:

Linearized BP

BP is approximated by

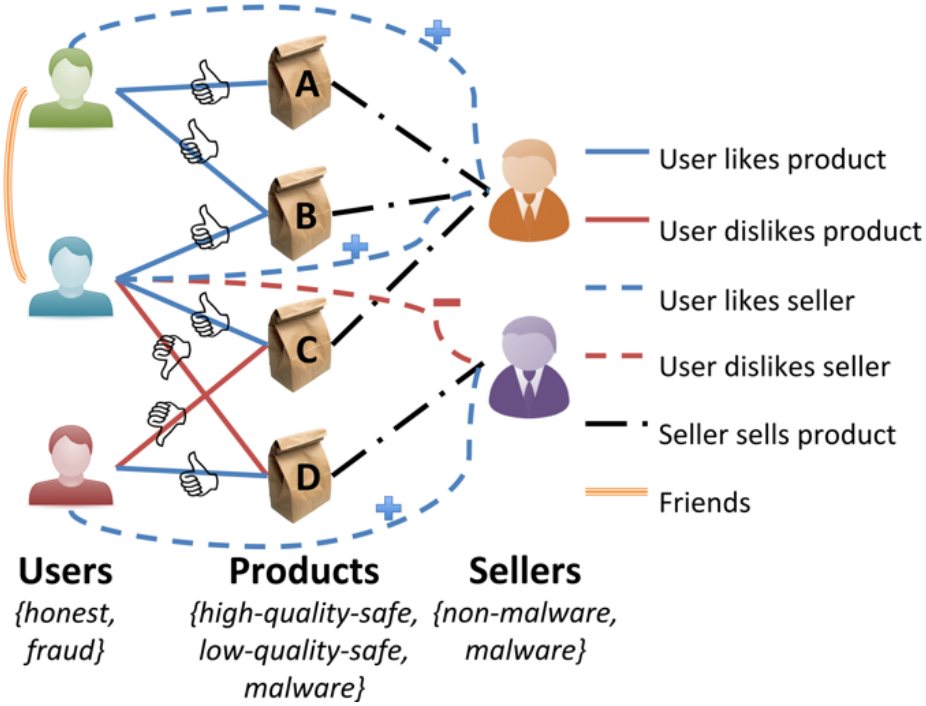
$$[\mathbf{I} + a\mathbf{D} - c'\mathbf{A}]\,\mathbf{b}_h = \phi_h$$

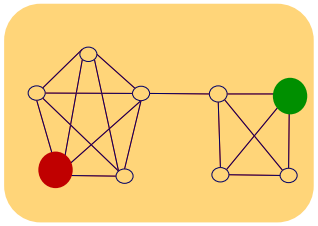
$$\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix} \quad \begin{bmatrix} d1 & & \\ & d2 & \\ & & d3 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad ? \quad \begin{bmatrix} 0 \\ -10^{-2} \\ 10^{-2} \end{bmatrix}$$

linear

- ✓ Closed-form formula?
- ✓ Convergence?

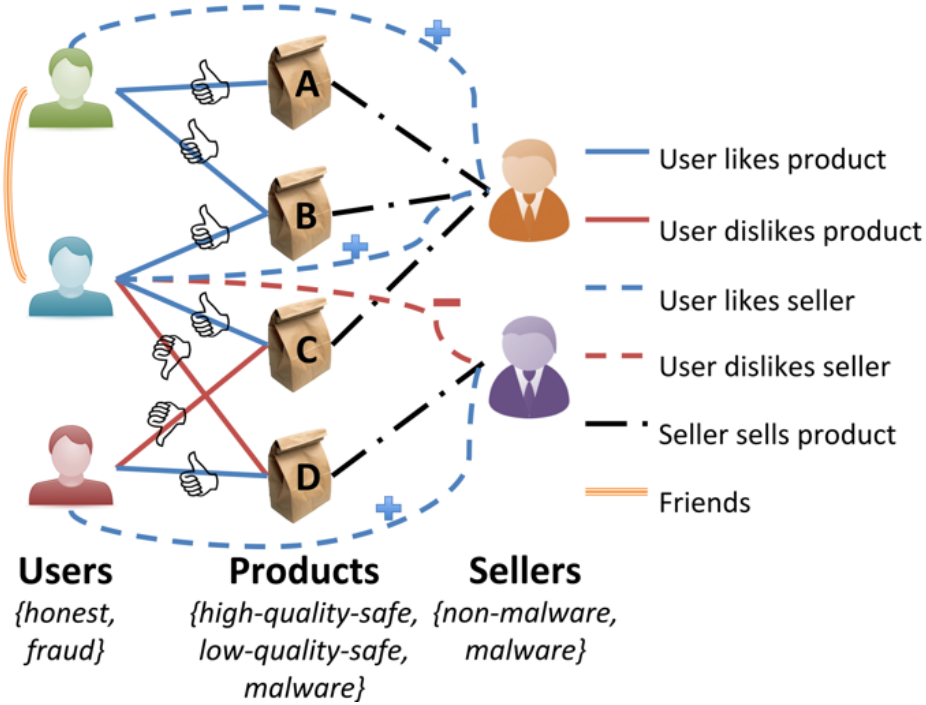
Problem: e-commerce ratings fraud





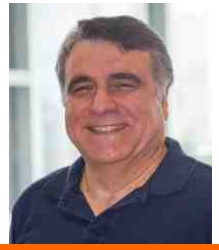
- **Given** a **heterogeneous** graph on users, products, sellers and positive/negative ratings with "seed labels"
- **Find** the top *k* most fraudulent users, products and sellers

Problem: e-commerce ratings fraud



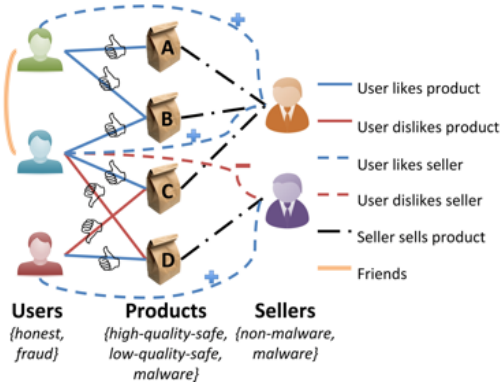
- **Given** a **heterogeneous** graph on users, products, sellers and positive/negative ratings with "seed labels"
- **Find** the top *k* most fraudulent users, products and sellers





Dhivya Eswaran, Stephan Günnemann, Christos Faloutsos, "*ZooBP:* Belief Propagation for Heterogeneous Networks", VLDB 2017

Problem: e-commerce ratings frau



Theorem 1 (ZooBP). *If* $\mathbf{b}, \mathbf{e}, \mathbf{P}, \mathbf{Q}$ *are constructed as described above, the linear equation system approximating the final node beliefs given by BP is:*

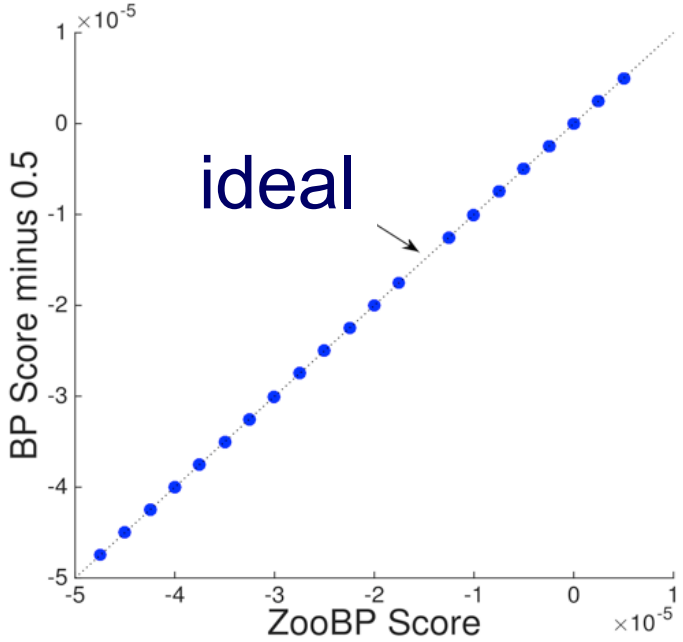
$$\mathbf{b} = \mathbf{e} + (\mathbf{P} - \mathbf{Q})\mathbf{b} \qquad (\text{ZooBP}) \qquad (10)$$

Dhivya Eswaran, Stephan Günnemann, Christos Faloutsos, "*ZooBP:* Belief Propagation for Heterogeneous Networks", VLDB 2017

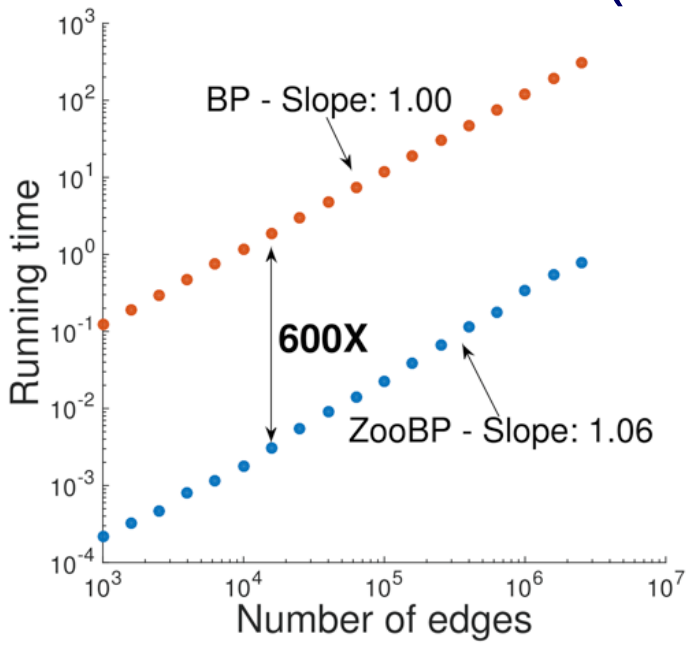
ZooBP: features

Fast; convergence guarantees.

600x (matlab)
3x (C++)



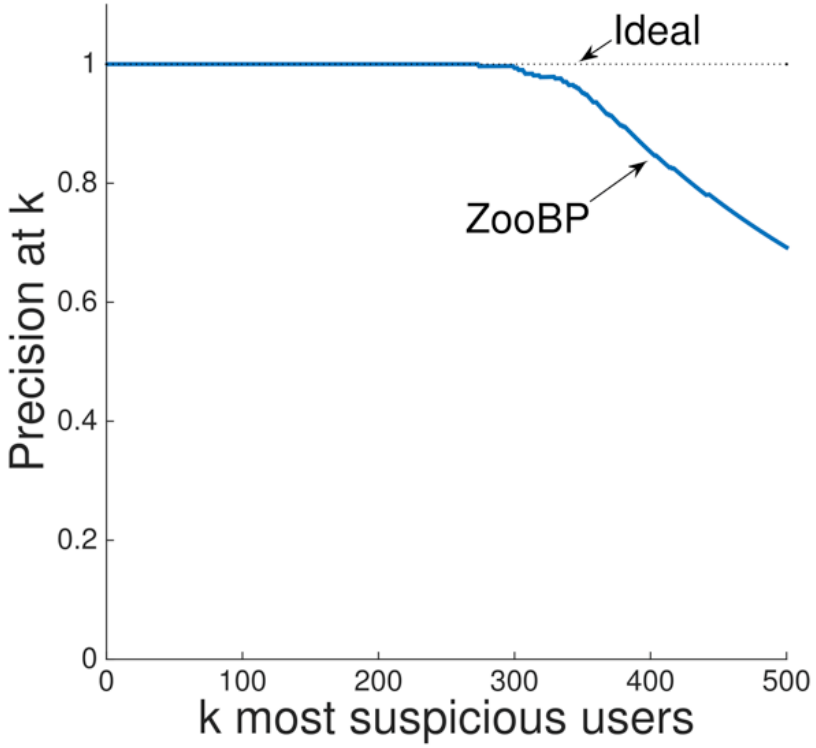
Near-perfect accuracy



linear in graph size

Dhivya Eswaran, Stephan Günnemann, Christos Faloutsos, "*ZooBP:* Belief Propagation for Heterogeneous Networks", VLDB 2017

ZooBP in the real world



- Near 100% precision on top 300 users (Flipkart)
- Flagged users: suspicious
  - 400 ratings in 1 sec
  - 5000 good ratings and no bad ratings

Dhivya Eswaran, Stephan Günnemann, Christos Faloutsos, "*ZooBP:* Belief Propagation for Heterogeneous Networks", VLDB 2017

ZooBP: code etc

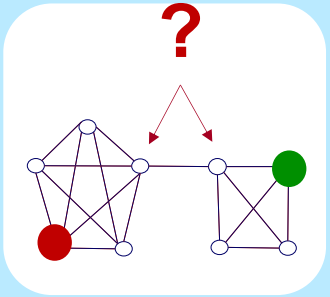
http://www.cs.cmu.edu/~deswaran/code/zoobp.zip

(now at Amazon,
Search Science & AI
deswaran@)

Dhivya Eswaran, Stephan Günnemann, Christos Faloutsos, "*ZooBP:* Belief Propagation for Heterogeneous Networks", VLDB 2017

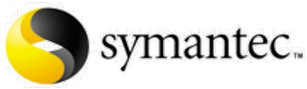
Bird's eye view

- Introduction – Motivation
- Part#1: (simple) Graphs
  - …
  - P1.4: belief propagation
    - Basics
    - Fast, linear approximation (FaBP)
    - Latest: zooBP
    - Success stories



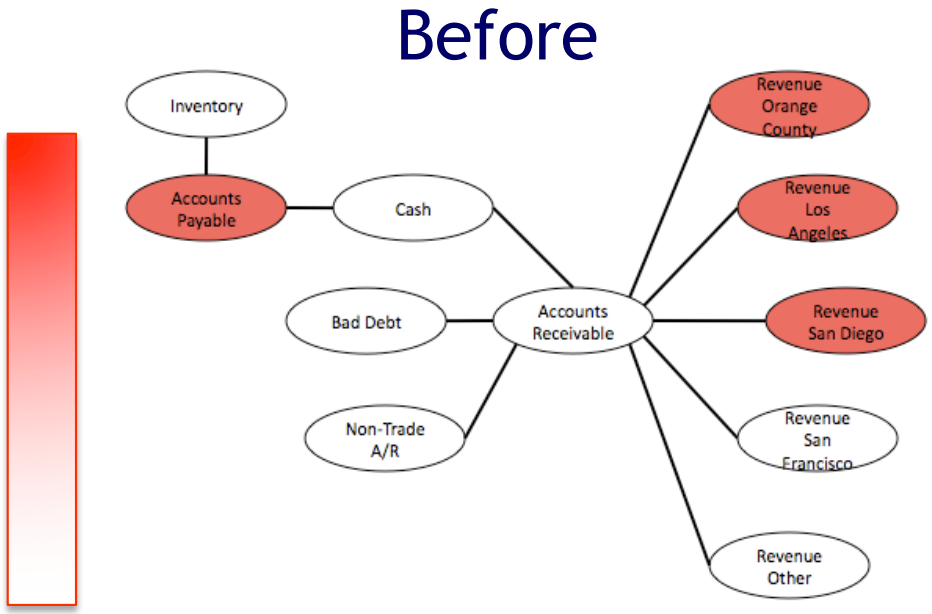
Other ‘success stories’?

- Accounting fraud
- Malware detection



Network Effect Tools: SNARE

- Some accounts are sort-of-suspicious – how to combine weak signals?



Mary McGlohon, Stephen Bay, Markus G. Anderle, David M. Steier, Christos Faloutsos: *SNARE: a link analytic system for graph labeling and risk detection*. KDD 2009: 1265-1274

Polonium: Tera-Scale Graph Mining and Inference for Malware Detection

SDM 2011, Mesa, Arizona

Polo Chau
Machine Learning Dept

symantec.

Carey Nachenberg
Vice President & Fellow

symantec.

Jeffrey Wilhelm
Principal Software Engineer

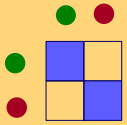
symantec.

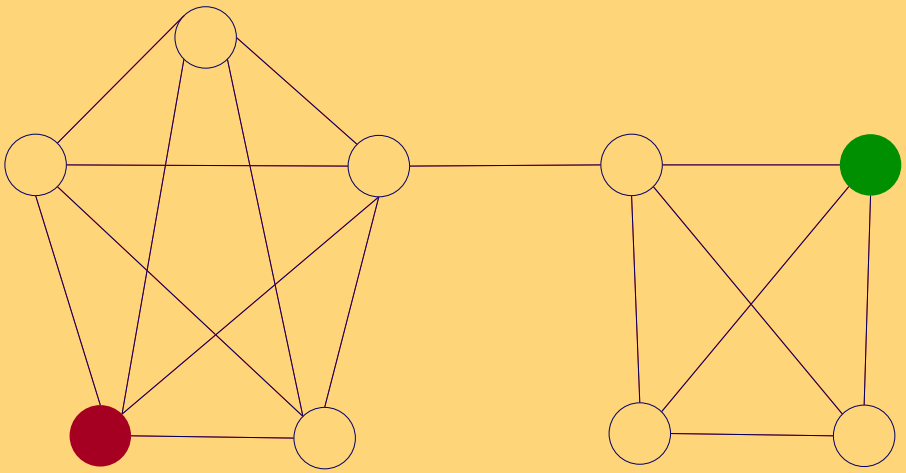
Adam Wright
Software Engineer

Prof. Christos Faloutsos
Computer Science Dept

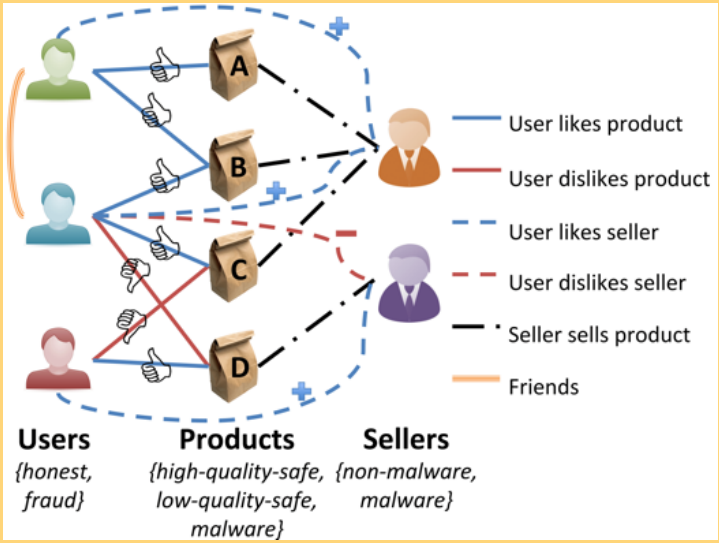
Short answer:

- What color, for the rest?
- A: Belief Propagation ('zooBP')





www.cs.cmu.edu/~deswaran/code/zoobp.zip



Bird's eye view

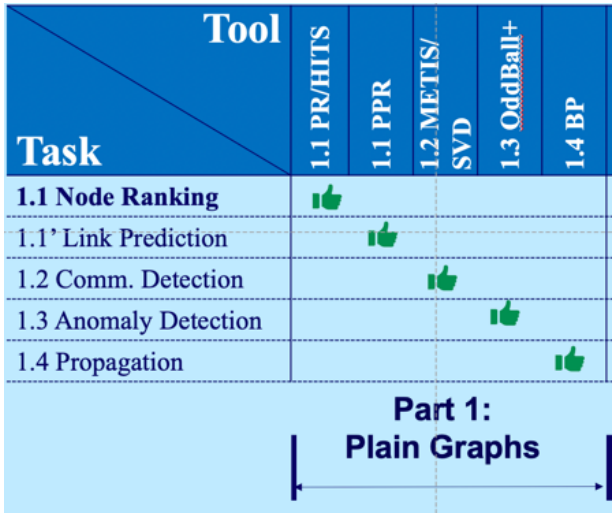
Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction		👍							
1.2 Comm. Detection			👍						
1.3 Anomaly Detection				👍					
1.4 Propagation					👍				

Part 1: Plain Graphs

Part 2: Complex Graphs

Conclusions for Part P1

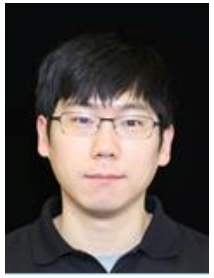
- Over-arching conclusion:
  - Many, time-tested tools for plain graphs (PR, SVD, BP)



Thanks to

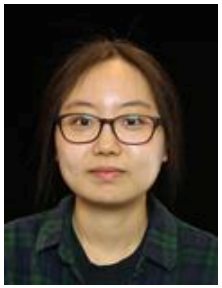
Danai Koutra
U. Michigan

Namyong Park
CMU



Dhivya Eswaran
CMU -> Amazon

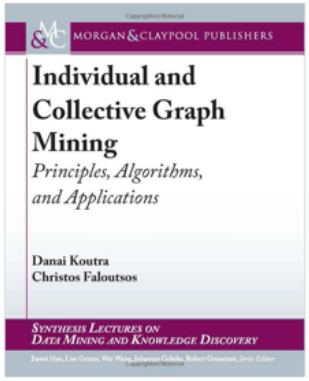
Hyun Ah Song
CMU -> Amazon



Vagelis
Papalexakis
UCR

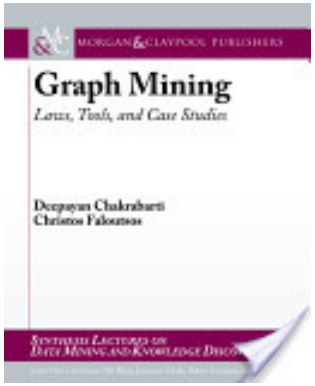
P1 – Graphs - More references

Danai Koutra and Christos Faloutsos,
Individual and Collective Graph Mining: Principles, Algorithms, and Applications
October 2017, Morgan Claypool



P1 – Graphs - More references

Deepayan Chakrabarti and Christos Faloutsos,
Graph Mining: Laws, Tools, and Case Studies
Oct. 2012, Morgan Claypool.



P1 – Graphs - More references

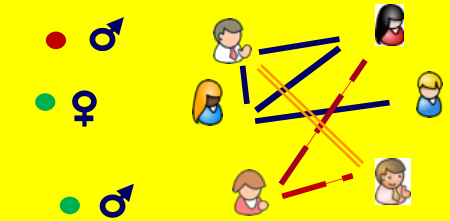
Anomaly detection

- Leman Akoglu, Hanghang Tong, & Danai Koutra, *Graph based anomaly detection and description: a survey* Data Mining and Knowledge Discovery (2015) 29: 626.

- Arxiv version: https://arxiv.org/abs/1404.4679

Bird's eye view

Task \ Tool	1.1 PR/HITS	1.1 PPR	1.2 METIS/ SVD	1.3 OddBall+	1.4 BP	2.1 FM	2.1 Tensor	2.2 HIN	2.3 SRL
1.1 Node Ranking	👍								
1.1' Link Prediction		👍							
1.2 Comm. Detection			👍						
1.3 Anomaly Detection				👍					
1.4 Propagation					👍				



Part 1: Plain Graphs

Part 2: Complex Graphs