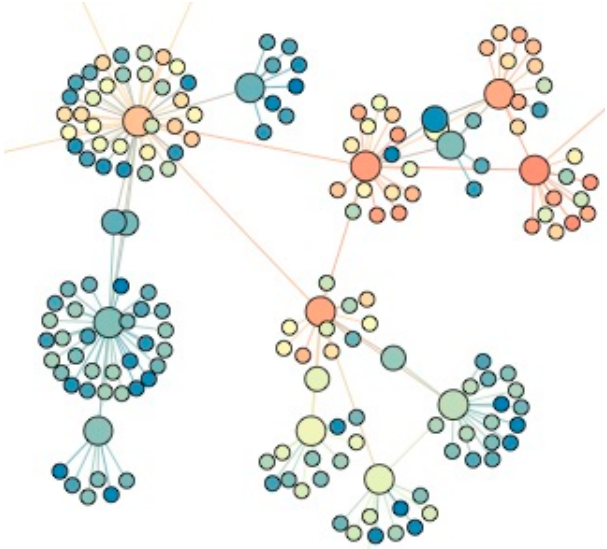


Link Analysis & Ranking

CS 224W

How to Organize the Web?

- **How to organize the Web?**
- **First try:** Human curated **Web directories**
  - Yahoo, DMOZ, LookSmart
- **Second try: Web Search**
  - **Information Retrieval** attempts to find relevant docs in a small and trusted set
    - Newspaper articles, Patents, etc.
  - **But:** Web is **huge**, full of untrusted documents, random things, web spam, etc.
  - **So we need a good way to rank webpages!**

Web Search: 2 Challenges

2 challenges of web search:

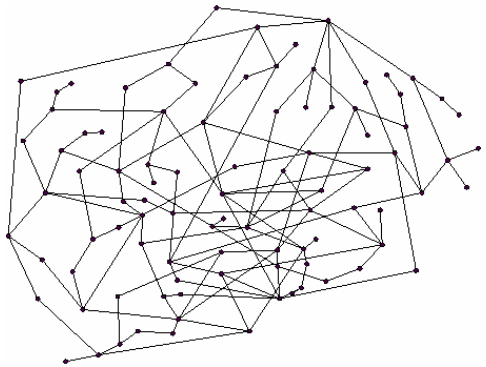
- **(1) Web contains many sources of information Who to "trust"?**
  - **Insight:** Trustworthy pages may point to each other!
- **(2) What is the "best" answer to query "newspaper"?**
  - No single right answer
  - **Insight:** Pages that actually know about newspapers might all be pointing to many newspapers

Ranking Nodes on the Graph

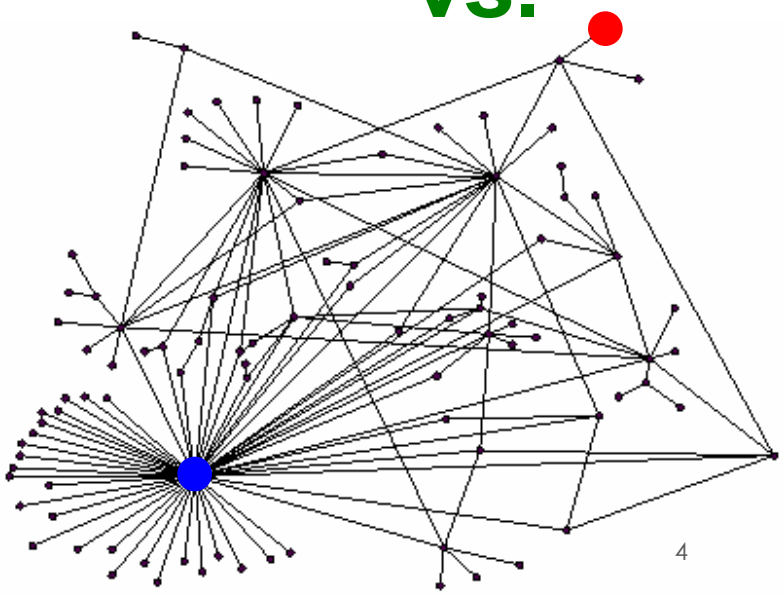
- **All web pages are not equally "important"**
  www.joe-schmoe.com vs. www.stanford.edu

- **We already know:**
  There is large diversity in the web-graph node connectivity.

- **So, let's rank the pages using the web graph link structure!**



vs.



Link Analysis Algorithms

- **We will cover the following Link Analysis approaches** to computing **importance** of nodes in a graph:
  - Hubs and Authorities (HITS)
  - Page Rank
  - ~~Topic-Specific (Personalized) Page Rank~~ <- another time

Sidenote: Various notions of **node centrality: Node u**

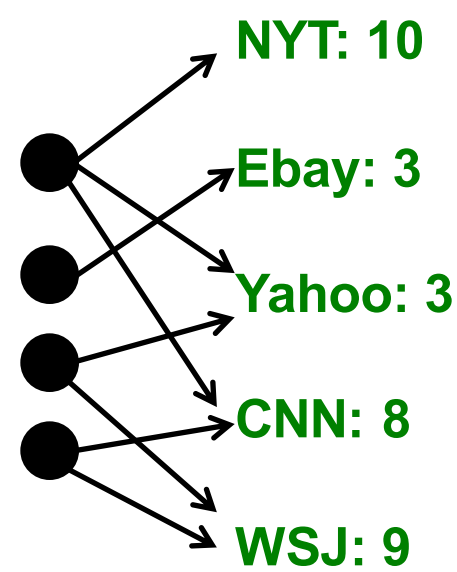
- **Degree centrality** = degree of u
- **Betweenness centrality** = #shortest paths passing through u
- **Closeness centrality** = avg. length of shortest paths from u to all other nodes of the network
- **Eigenvector centrality** = like PageRank

Hubs and Authorities

Link Analysis

- **Goal** (back to the newspaper example):
  - Don't just find newspapers. Find "experts" – pages that link in a coordinated way to good newspapers
- **Idea: Links as votes**
  - **Page is more important if it has more links**
    - In-coming links? Out-going links?
- **Hubs and Authorities**

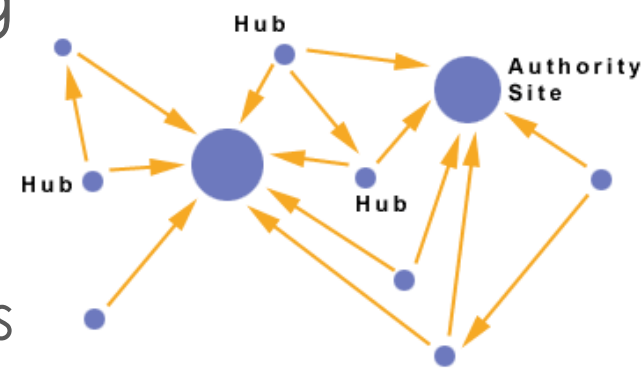
  Each page has **2** scores:
  - **Quality as an expert (hub):**
    - Total sum of votes of pages pointed to
  - **Quality as an content (authority):**
    - Total sum of votes of experts
  - **Principle of repeated improvement**



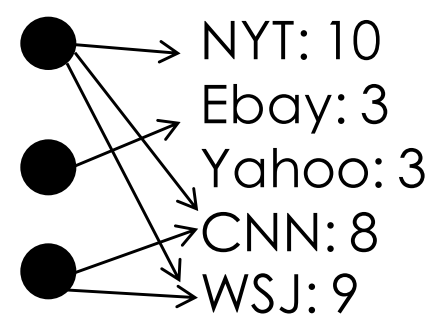
Hubs and Authorities

Interesting pages fall into two classes:

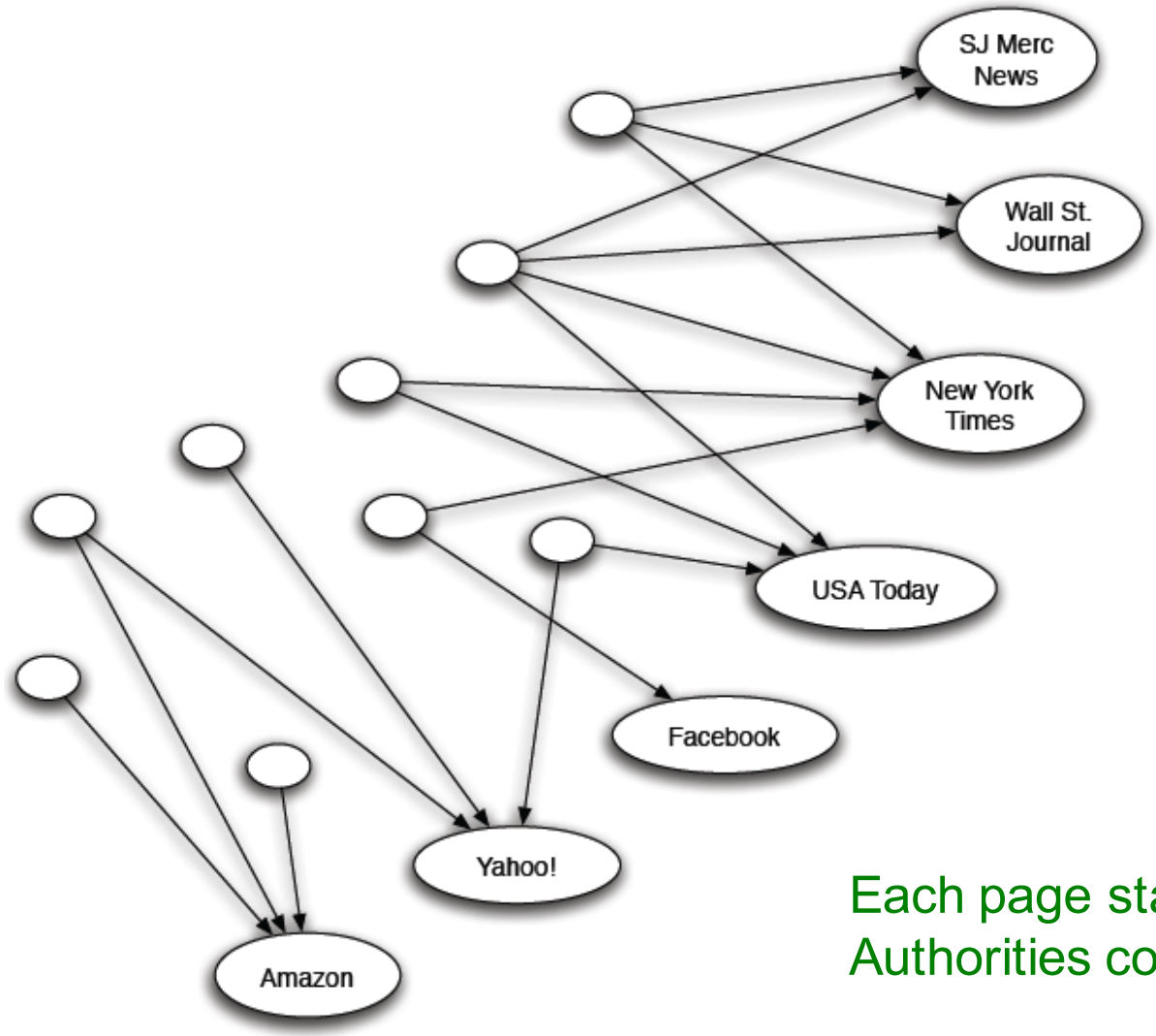
1. **Authorities** are pages containing useful information
   - Newspaper home pages
   - Course home pages
   - Home pages of auto manufacturers



2. **Hubs** are pages that link to authorities
   - List of newspapers
   - Course bulletin
   - List of U.S. auto manufacturers



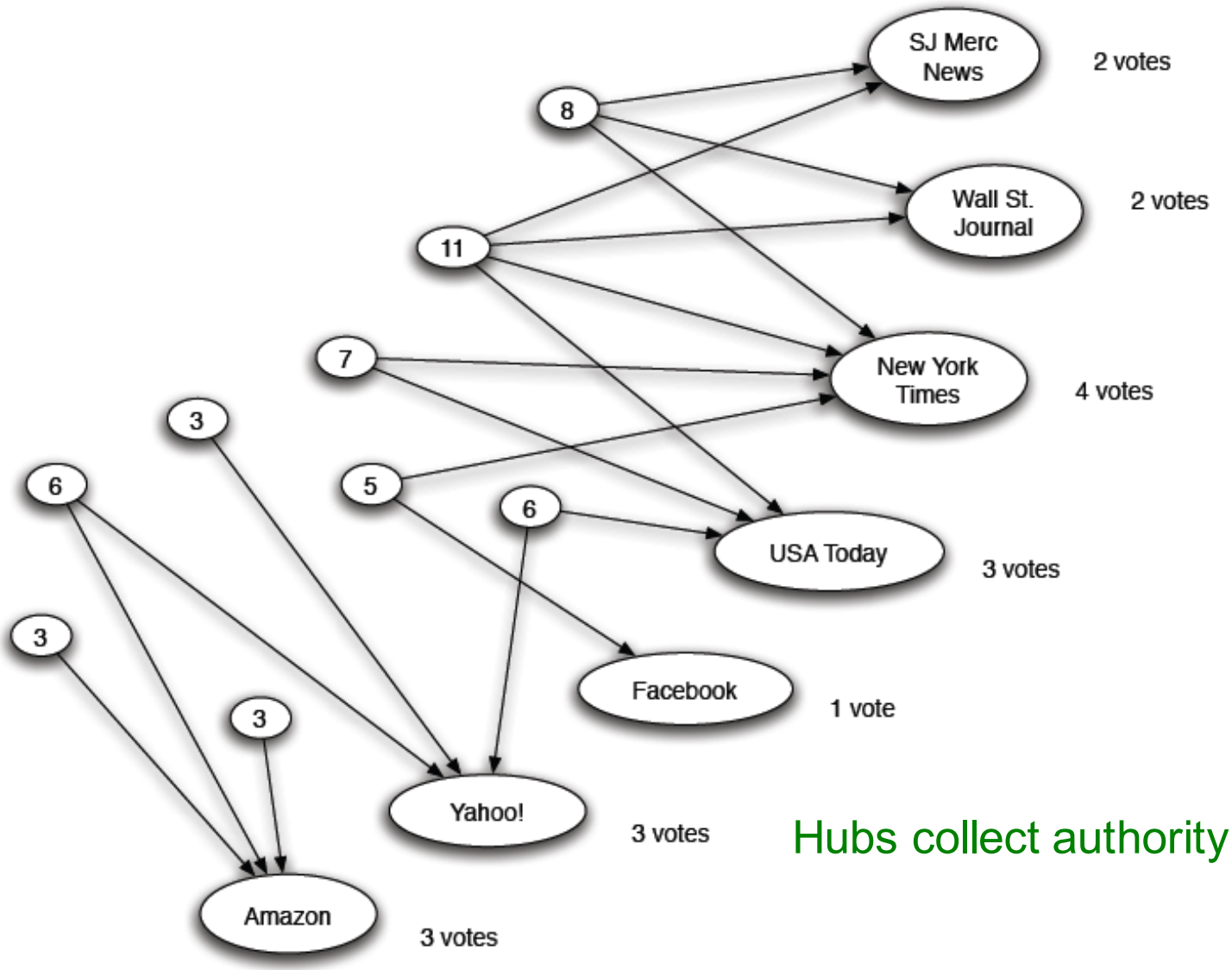
Counting in-links: Authority



Each page starts with **hub score 1**
Authorities collect their votes

(Note this is idealized example. In reality graph is not bipartite and each page has both a hub and the authority score)

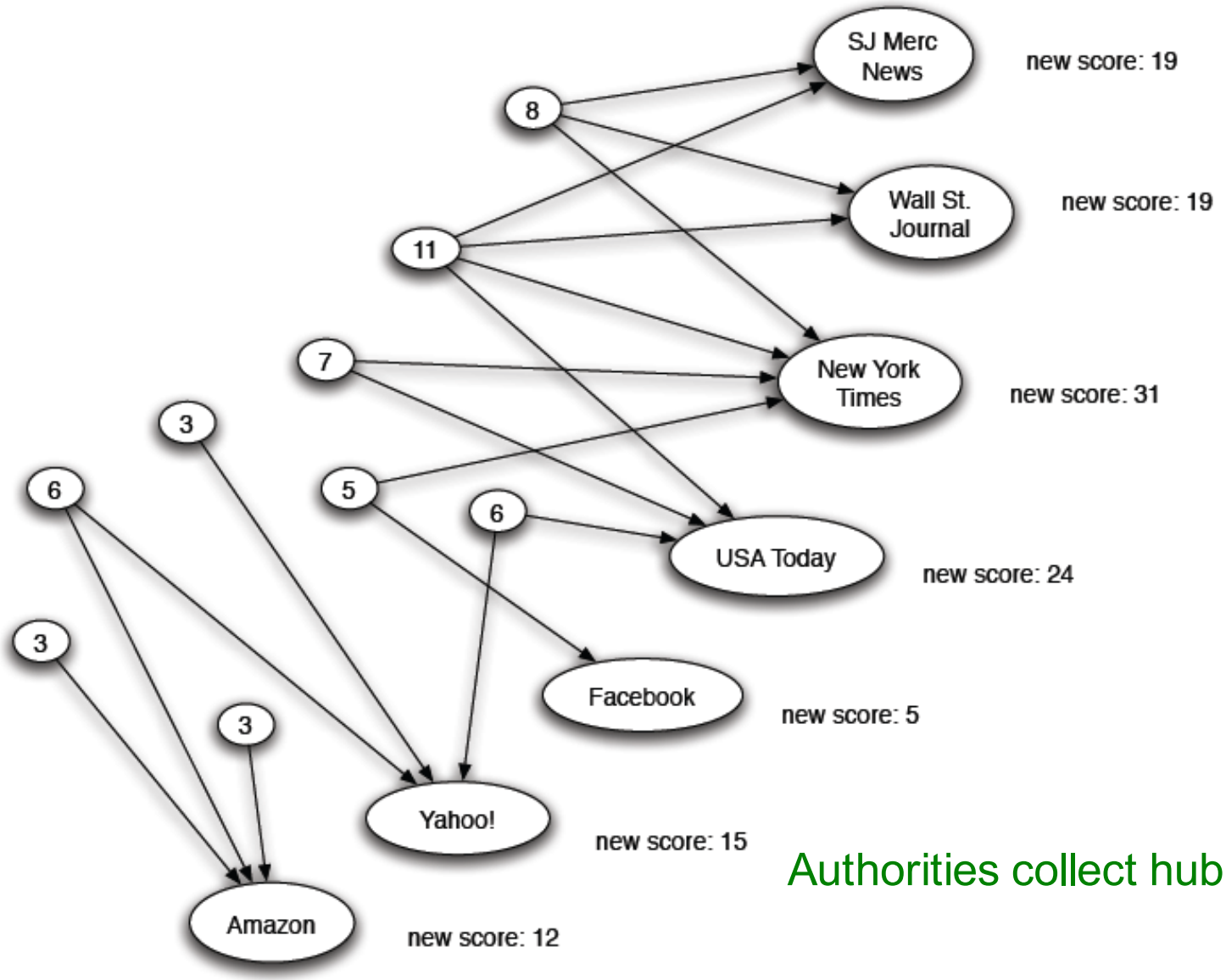
Expert Quality: Hub



Hubs collect authority scores

(Note this is idealized example. In reality graph is not bipartite and each page has both a hub and authority score)

Reweighting



Authorities collect hub scores

(Note this is idealized example. In reality graph is not bipartite and each page has both a hub and authority score)

Mutually Recursive Definition

- **A good hub links to many good authorities**
- **A good authority is linked from many good hubs**
  - Note a self-reinforcing recursive definition
- **Model using two scores for each node:**
  - **Hub** score and **Authority** score
  - Represented as vectors $\boldsymbol{h}$ and $\boldsymbol{a}$, where the i-th element is the hub/authority score of the i-th node

Hubs and Authorities

- **Each page i has 2 scores:**
  - Authority score: $\boldsymbol{a_i}$
  - Hub score: $\boldsymbol{h_i}$

Convergence criteria:

$$\sum_i \left(h_i^{(t)} - h_i^{(t+1)}\right)^2 < \varepsilon$$

$$\sum_i \left(a_i^{(t)} - a_i^{(t+1)}\right)^2 < \varepsilon$$

HITS algorithm:

- Initialize: $a_j^{(0)} = 1/\sqrt{\mathrm{n}}, \;\; \mathrm{h_j}^{(0)} = 1/\sqrt{\mathrm{n}}$
- Then keep iterating until **convergence**:
  - $\forall \boldsymbol{i}$: Authority: $a_i^{(t+1)} = \sum_{j \to i} h_j^{(t)}$
  - $\forall \boldsymbol{i}$: Hub: $h_i^{(t+1)} = \sum_{i \to j} a_j^{(t)}$
  - $\forall \boldsymbol{i}$: Normalize:

    $\sum_i \left(a_i^{(t+1)}\right)^2 = 1, \sum_j \left(h_j^{(t+1)}\right)^2 = 1$

Details!

Hubs and Authorities

- **Hits in the vector notation:**
  - Vector $\boldsymbol{a} = (\boldsymbol{a_1} \ldots, \boldsymbol{a_n})$, $\boldsymbol{h} = (\boldsymbol{h_1} \ldots, \boldsymbol{h_n})$
  - Adjacency matrix $\boldsymbol{A}$ (n x n): $\boldsymbol{A_{ij}} = \boldsymbol{1}$ if $\boldsymbol{i} \rightarrow \boldsymbol{j}$
- **Can rewrite** $\boldsymbol{h_i} = \sum_{\boldsymbol{i} \rightarrow \boldsymbol{j}} \boldsymbol{a_j}$ **as** $\boldsymbol{h_i} = \sum_{\boldsymbol{j}} \boldsymbol{A_{ij}} \cdot \boldsymbol{a_j}$
- **So:** $\boldsymbol{h} = \boldsymbol{A} \cdot \boldsymbol{a}$ **And similarly:** $\boldsymbol{a} = \boldsymbol{A^T} \cdot \boldsymbol{h}$
- **Repeat until convergence:**
  - $h^{(t+1)} = A \cdot a^{(t)}$
  - $a^{(t+1)} = A^T \cdot h^{(t)}$
  - Normalize $a^{(t+1)}$ and $h^{(t+1)}$

Details!

Hubs and Authorities

- What is $\boldsymbol{a} = \boldsymbol{A}^T \cdot \boldsymbol{h}$?
- Then: $\boldsymbol{a} = \underbrace{\boldsymbol{A}^T \cdot \underbrace{(\boldsymbol{A} \cdot \boldsymbol{a})}_{\text{new } \boldsymbol{h}}}_{\text{new } \boldsymbol{a}}$
- $\boldsymbol{a}$ is updated (in 2 steps):

$$a = A^T (A\, a) = (A^T A)\, a$$

- $\boldsymbol{h}$ is updated (in 2 steps)

$$h = A\,(A^T h) = (A\, A^T)\, h$$

- Thus, in $2\boldsymbol{k}$ steps:

$$a = (A^T \cdot A)^k \cdot a$$

$$h = (A \cdot A^T)^k \cdot h$$

Repeated matrix powering

Hubs and Authorities

- **<u>Definition:</u> Eigenvectors & Eigenvalues**
- Let $\boldsymbol{R} \cdot \boldsymbol{x} = \boldsymbol{\lambda} \cdot \boldsymbol{x}$
  for some scalar $\boldsymbol{\lambda}$, vector $\boldsymbol{x}$, matrix $\boldsymbol{R}$
  - Then $\boldsymbol{x}$ **is an eigenvector**, and $\boldsymbol{\lambda}$ is its **eigenvalue**
- **The <u>steady state</u> (HITS has converged):**
  - $\boldsymbol{A}^{\boldsymbol{T}} \cdot \boldsymbol{A} \cdot \boldsymbol{a} = \boldsymbol{c}' \cdot \boldsymbol{a}$
  - $\boldsymbol{A} \cdot \boldsymbol{A}^{\boldsymbol{T}} \cdot \boldsymbol{h} = \boldsymbol{c}'' \cdot \boldsymbol{h}$
- **So, authority $\boldsymbol{a}$** is eigenvector of $\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}$
  (associated with the largest eigenvalue)
  Similarly: **hub $\boldsymbol{h}$** is eigenvector of $\boldsymbol{A}\boldsymbol{A}^{\boldsymbol{T}}$

Note constants *c',c"* don't matter as we normalize them out every step of HITS

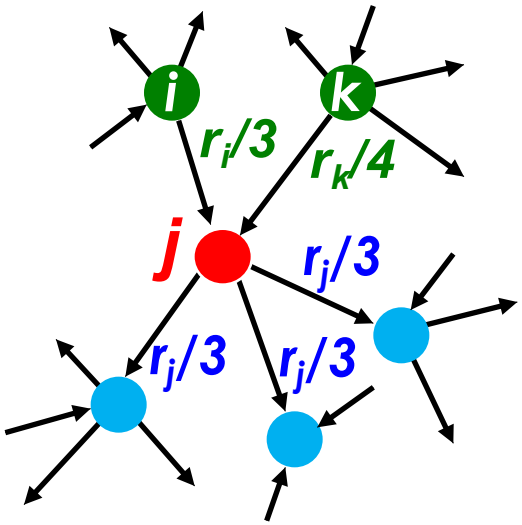
PageRank

Links as Votes

- **Still the same idea: Links as votes**
  - **Page is more important if it has more links**
    - In-coming links? Out-going links?
- **Think of in-links as votes:**
  - www.stanford.edu has 23,400 in-links
  - www.joe-schmoe.com has 1 in-link
- **Are all in-links equal?**
  - Links from important pages count more
  - Recursive question!

PageRank: The "Flow" Model

- **A "vote" from an important page is worth more:**
  - Each link's vote is proportional to the **importance** of its source page
  - If page ***i*** with importance r_i has d_i out-links, each link gets r_i / d_i votes
  - Page ***j***'s own importance r_j is the sum of the votes on its in-links



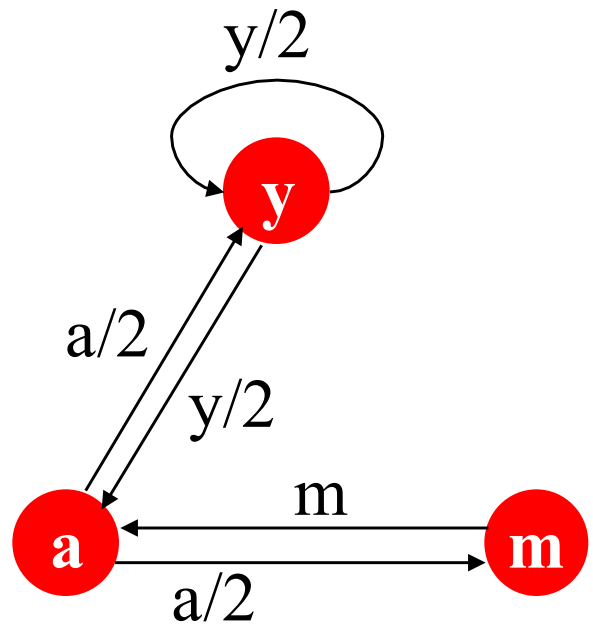
$$r_j = r_i/3 + r_k/4$$

PageRank: The "Flow" Model

- A page is important if it is pointed to by other important pages
- Define a "rank" r_j for node j

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

d_i … out-degree of node i



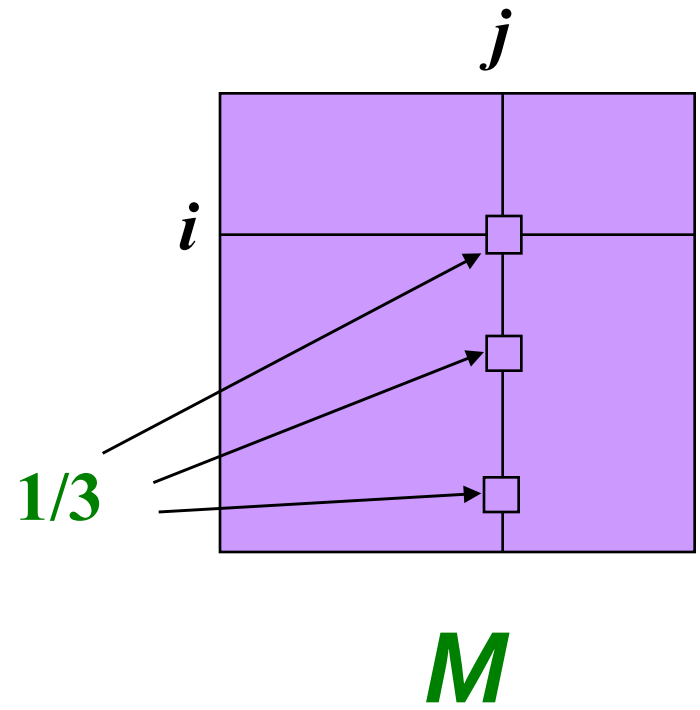
"Flow" equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Why do we divide by out-degree?

PageRank: Matrix Formulation

- **Stochastic adjacency matrix M**
  - Let page j have d_j out-links
  - If $j \rightarrow i$, then $M_{ij} = \frac{1}{d_j}$
    - M is a **column stochastic matrix**
      - **Columns** sum to **1**



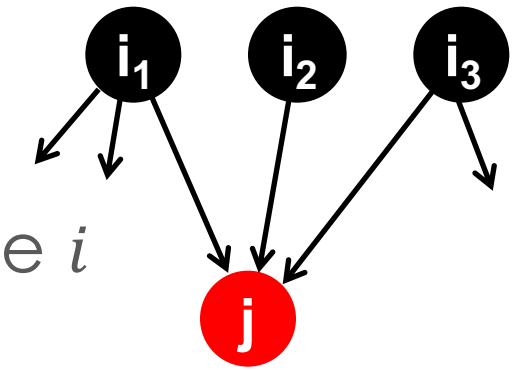
- **Rank vector r:** An entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$

- **The flow equations can be written**

$$r = M \cdot r$$

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

Random Walk Interpretation



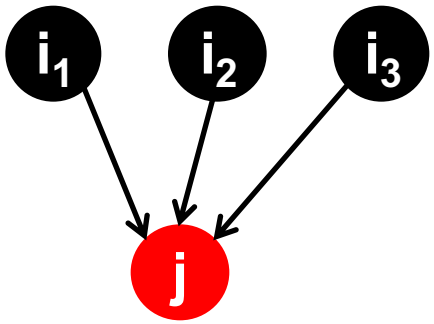
- **Imagine a random web surfer:**
  - At any time $\boldsymbol{t}$, surfer is on some page i
  - At time $\boldsymbol{t+1}$, the surfer follows an out-link from $\boldsymbol{i}$ uniformly at random
  - Ends up on some page $\boldsymbol{j}$ linked from $\boldsymbol{i}$
  - Process repeats indefinitely

$$r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d}_{\mathrm{out}}(\mathrm{i})}$$

- **Let:**
  - $\boldsymbol{p}(\boldsymbol{t})$ … vector whose i^{th} coordinate is the prob. that the surfer is at page i at time t
  - So, $\boldsymbol{p}(\boldsymbol{t})$ is a probability distribution over pages

The Stationary Distribution

- **Where is the surfer at time *t+1*?**
  - Follows a link uniformly at random



$$p(t+1) = \mathrm{M} \cdot p(t)$$

$$\boldsymbol{p}(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{M} \cdot \boldsymbol{p}(\boldsymbol{t})$$

- Suppose the random walk reaches a state
  $\boldsymbol{p}(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{M} \cdot \boldsymbol{p}(\boldsymbol{t}) = \boldsymbol{p}(\boldsymbol{t})$
  then $\boldsymbol{p}(t)$ is **stationary distribution** of a random walk
- **Our original rank vector** $\boldsymbol{r}$ satisfies $\boldsymbol{r} = \boldsymbol{M} \cdot \boldsymbol{r}$
  - **So, $\boldsymbol{r}$ is a stationary distribution for the random walk**

Computing PageRank

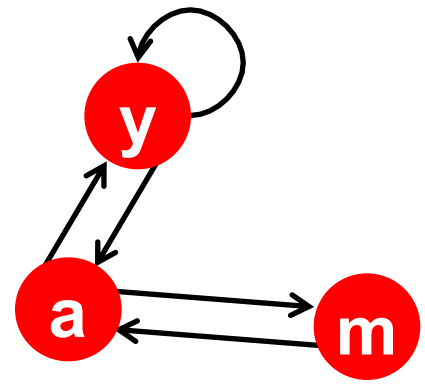
Given a web graph with *n* nodes, where the nodes are pages and edges are hyperlinks

- Assign each node an initial page rank
  - Repeat until convergence ($\Sigma_i \, | r_i^{(t+1)} - r_i^{(t)} | < \varepsilon$)
    - Calculate the page rank of each node

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_i}$$

d_i …. **out-degree of node** i

Computing PageRank



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

- **Power Iteration:**
  - Set $r_j \leftarrow 1/N$
  - **1:** $r'_j \leftarrow \sum_{i \to j} \frac{r_i}{d_i}$
  - **2:** $r \leftarrow r'$
  - If $|r - r'| > \varepsilon$: goto **1**

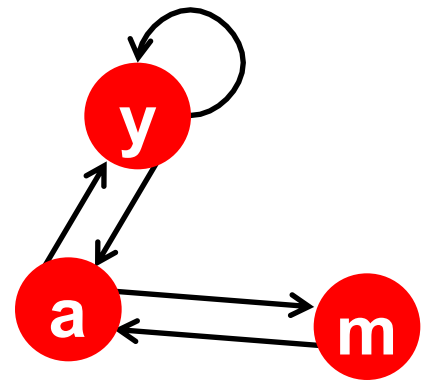
$$\mathbf{r_y = r_y/2 + r_a/2}$$
$$\mathbf{r_a = r_y/2 + r_m}$$
$$\mathbf{r_m = r_a/2}$$

- **Example:**

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{matrix} 1/3 \\ 1/3 \\ 1/3 \end{matrix}$$

Iteration 0, 1, 2, …

Computing PageRank



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

- **Power Iteration:**
  - Set $r_j \leftarrow 1/N$
  - **1:** $r'_j \leftarrow \sum_{i \to j} \frac{r_i}{d_i}$
  - **2:** $r \leftarrow r'$
  - If $|r - r'| > \varepsilon$: goto **1**

$$\mathbf{r_y = r_y/2 + r_a/2}$$
$$\mathbf{r_a = r_y/2 + r_m}$$
$$\mathbf{r_m = r_a/2}$$

- **Example:**

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{matrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \ldots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 3/15 \end{matrix}$$

Iteration 0, 1, 2, …

PageRank: Three Questions

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d_i}}$$

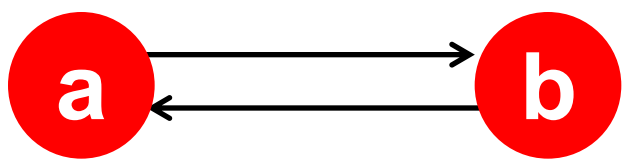
or equivalently

$$r = Mr$$

- Does this converge?
- Does it converge to what we want?
- Are results reasonable?

Does this converge?

- The "Spider trap" problem:



$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

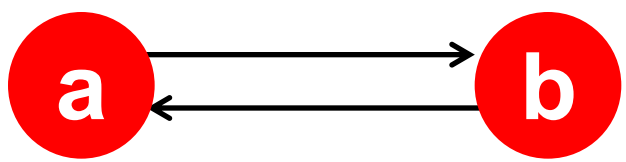
- Example:

Iteration: 0, 1, 2, 3…

$$\begin{matrix} r_a \\ r_b \end{matrix} = \begin{matrix} 1 \\ 0 \end{matrix} \Bigg| \quad ?$$

Does this converge?

- The "Spider trap" problem:



$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

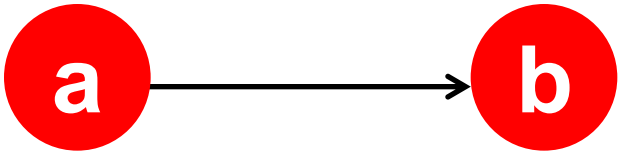
- Example:

Iteration: 0, 1, 2, 3…

r_a =	1	0	1	0
r_b	0	1	0	1

Dead end problem

- The "Dead end" problem:



$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d_i}}$$

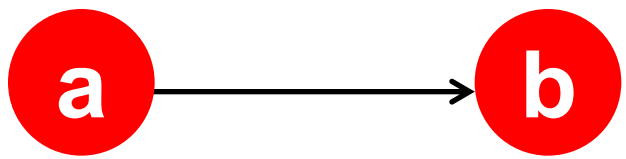
- Example:

Iteration: 0, 1, 2, 3…

$$\begin{matrix} r_a \\ r_b \end{matrix} = \begin{matrix} 1 \\ 0 \end{matrix} \Bigg| \quad ?$$

Does it converge to what we want?

- The "Dead end" problem:



$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d_i}}$$

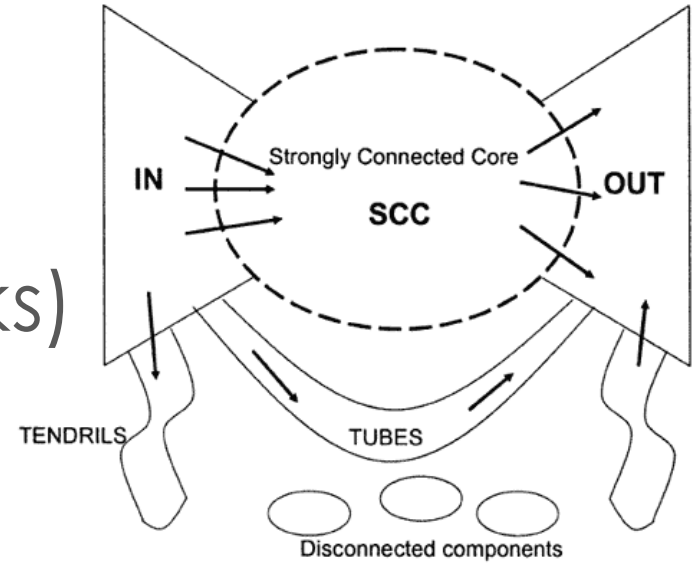
- Example:

	Iteration: 0,	1,	2,	3…
r_a =	1	0	0	0
r_b	0	1	0	0

RageRank: Problems

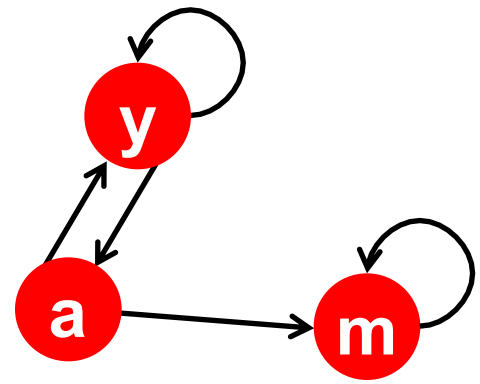
2 problems:

- **(1)** Some pages are **dead ends** (have no out-links)
  - Such pages cause importance to "leak out"



- **(2) Spider traps** (all out-links are within the group)
  - Eventually spider traps absorb all importance

Problem: Spider Traps



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	1

- **Power Iteration:**
  - Set $r_j = \frac{1}{N}$
  - $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
    - And iterate

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2 + r_m$$

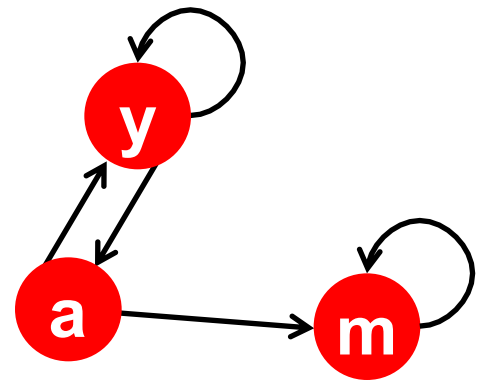
- **Example:**

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{matrix} 1/3 \\ 1/3 \\ 1/3 \end{matrix} \Bigg| \; ?$$

Iteration 0, 1, 2, …

Problem: Spider Traps

- **Power Iteration:**
  - Set $r_j = \frac{1}{N}$
  - $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
    - And iterate



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	1

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2 + r_m$$

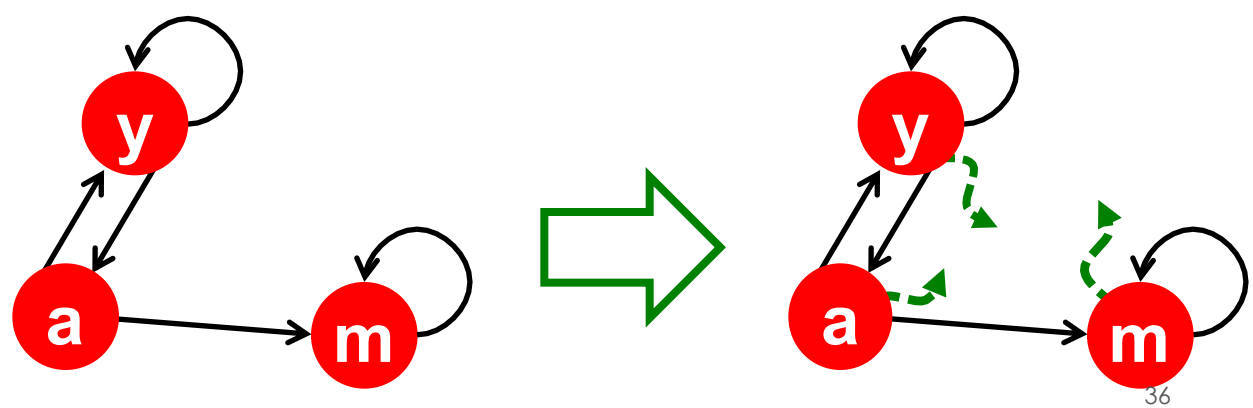
- **Example:**

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{matrix} 1/3 \\ 1/3 \\ 1/3 \end{matrix} \Bigg| \begin{matrix} 2/6 \\ 1/6 \\ 3/6 \end{matrix} \Bigg| \begin{matrix} 3/12 \\ 2/12 \\ 7/12 \end{matrix} \Bigg| \begin{matrix} 5/24 \\ 3/24 \\ 16/24 \end{matrix} \quad \ldots \quad \begin{matrix} 0 \\ 0 \\ 1 \end{matrix}$$

Iteration 0, 1, 2, …

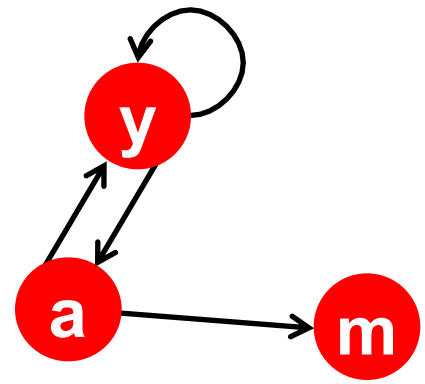
Solution: Random Teleports

- The Google solution for spider traps: **At each time step, the random surfer has two options**
  - With prob. β, follow a link at random
  - With prob. **1-β**, jump to a random page
  - Common values for β are in the range 0.8 to 0.9
- **Surfer will teleport out of spider trap within a few time steps**



Problem: Dead Ends

- **Power Iteration:**
  - Set $r_j = \frac{1}{N}$
  - $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
    - And iterate



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2$$

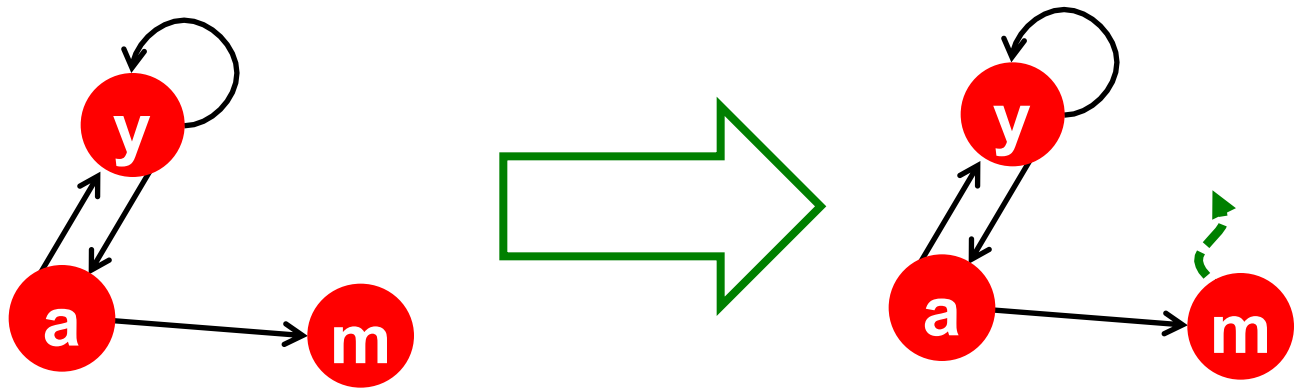
- **Example:**

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{matrix} 1/3 \\ 1/3 \\ 1/3 \end{matrix} \Bigg| \begin{matrix} 2/6 \\ 1/6 \\ 1/6 \end{matrix} \Bigg| \begin{matrix} 3/12 \\ 2/12 \\ 1/12 \end{matrix} \Bigg| \begin{matrix} 5/24 \\ 3/24 \\ 2/24 \end{matrix} \quad \ldots \quad \begin{matrix} 0 \\ 0 \\ 0 \end{matrix}$$

Iteration 0, 1, 2, …

Solution: escape dead ends immediately

- **Teleports:** Follow random teleport links with probability **1.0** from dead-ends
  - Adjust matrix accordingly



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

	y	a	m
y	½	½	⅓
a	½	0	⅓
m	0	½	⅓

Final PageRank Equation

- **Google's solution: At each step, random surfer has two options:**
  - With probability β, follow a link at random
  - With probability $1-\beta$, jump to some random page
- **PageRank equation** [Brin-Page, '98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{n}$$

d_i … out-degree of node i

The above formulation assumes that $\boldsymbol{M}$ has no dead ends. We can either preprocess matrix $\boldsymbol{M}$ **(bad!)** or explicitly follow random teleport links with probability 1.0 from dead-ends. See P. Berkhin, *A Survey on PageRank Computing*, Internet Mathematics, 2005.

PageRank & Eigenvectors

Details!

- **PageRank as a principal eigenvector**

$$\boldsymbol{r} = \boldsymbol{M} \cdot \boldsymbol{r} \quad \text{or equivalently } \boldsymbol{r}_j = \sum_i \frac{r_i}{d_i}$$

- **But we really want (**):**

$$r_j = \beta \sum_i \frac{r_i}{d_i} + (1-\beta)\frac{1}{n}$$

d_i … out-degree of node i

- **Let's define:**

$$M'_{ij} = \beta\, M_{ij} + (1-\beta)\frac{1}{n}$$

- **Now we get what we want:**

$$\boldsymbol{r} = \boldsymbol{M}' \cdot \boldsymbol{r}$$

Note: M is a sparse matrix but $\boldsymbol{M}'$ is dense (all entries $\neq 0$). In practice we never "materialize" M but rather we use the "sum" formulation (**)

- **What is $(1-\beta)$?**
  - In practice 0.15 (Jump approx. every 5-6 links)

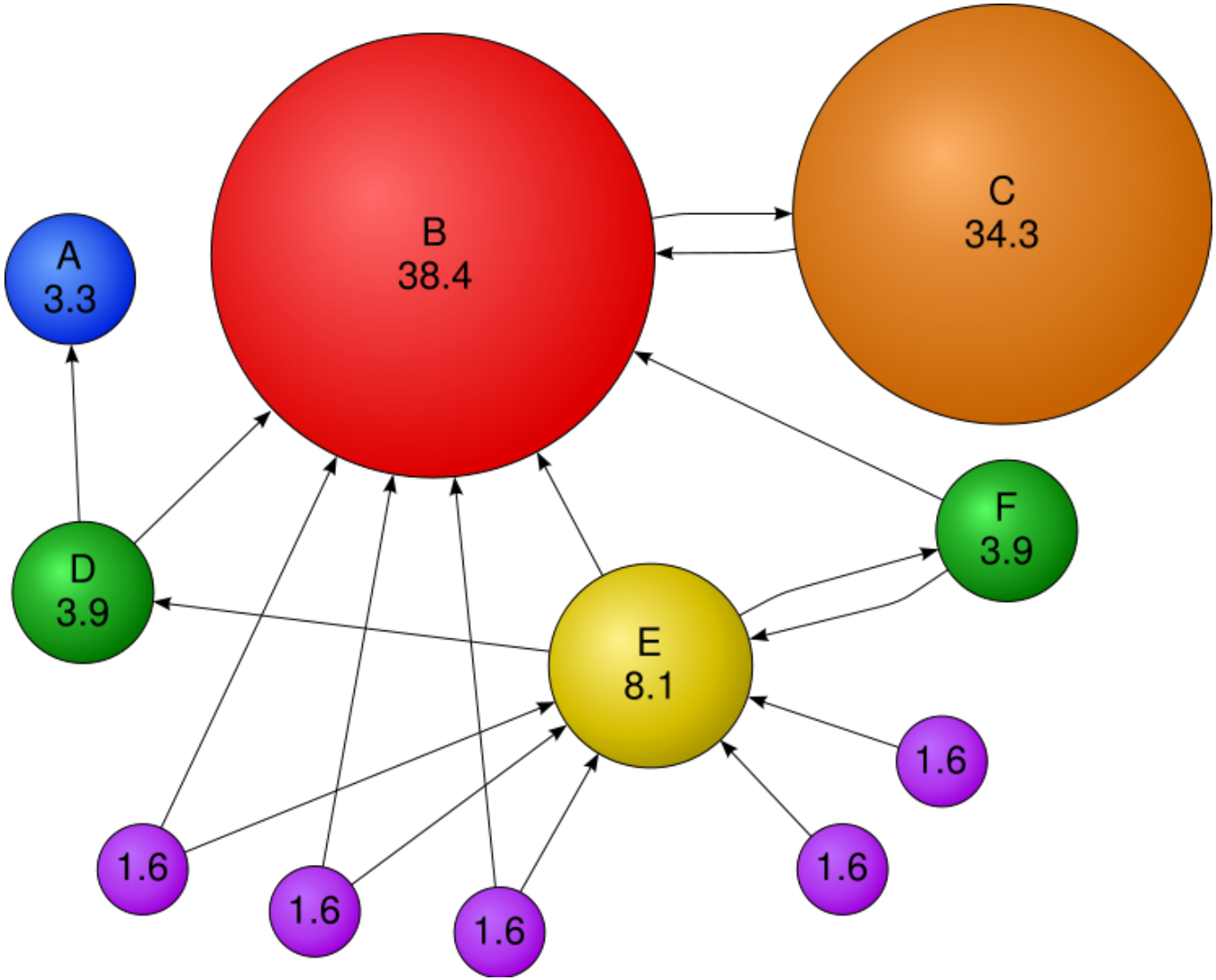
The PageRank Algorithm

- **Input:** Graph G and parameter β
  - Directed graph G with **spider traps** and **dead ends**
  - Parameter β
- **Output:** PageRank vector r
  - **Set:** $r_j^{(0)} = \frac{1}{N}, \quad t = 1$
  - **do:**
    - $\forall j: r'^{(t)}_j = \sum_{i \to j} \beta \frac{r_i^{(t-1)}}{d_i}$

      $r'^{(t)}_j = 0$ if in-deg. of j is 0
    - **Now re-insert the leaked PageRank:**

      $\forall j: r_j^{(t)} = r'^{(t)}_j + \frac{1-S}{N}$ **where:** $S = \sum_j r'^{(t)}_j$
    - $t = t + 1$
  - **while** $\sum_j \left| r_j^{(t)} - r_j^{(t-1)} \right| > \varepsilon$

Example



PageRank and HITS

- **PageRank and HITS are two solutions to the same problem:**
  - **What is the value of an in-link from *u* to *v*?**
  - In the PageRank model, the value of the link depends on the links **into** *u*
  - In the HITS model, it depends on the value of the other links **out of** *u*

- **The destinies of PageRank and HITS post-1998 were very different**

Example application: expertise ranking

Oozing out knowledge

NAVER 지식iN 지식과 내가 함께 커가는 곳

Knowledge In

``Knowledge search is like oozing out knowledge in human brains to the Internet. People who know something better than others can present their know-how, skills or knowledge"

NHN CEO Chae Hwi-young

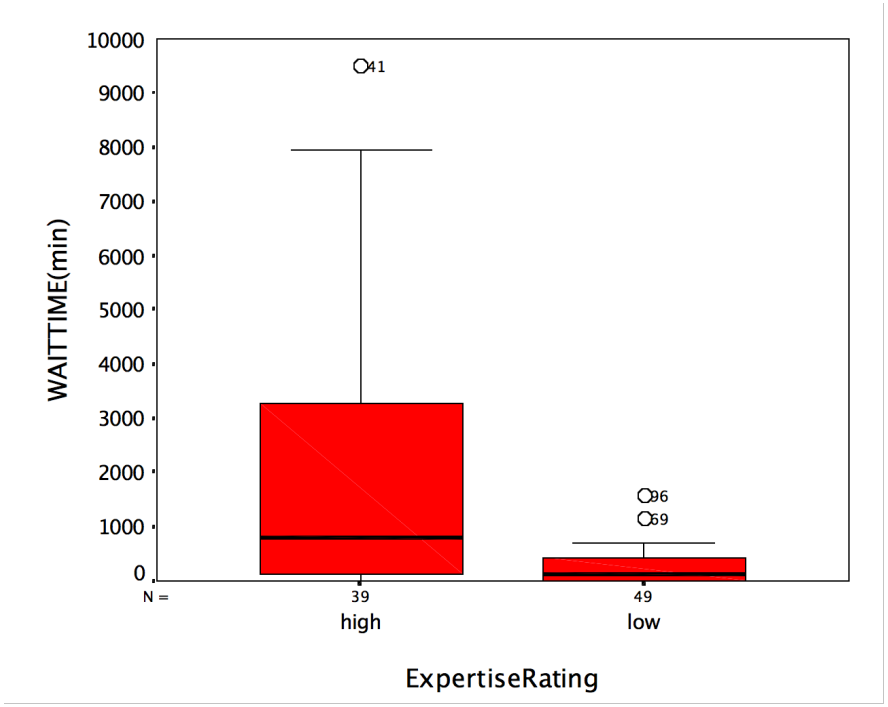
*"(It is) the **next generation of search**… (it) is a kind of **collective brain** -- a searchable database of everything everyone knows. It's a **culture of generosity**. The fundamental belief is that **everyone knows something**."*

-- Eckart Walther (Yahoo Research)

YAHOO! ANSWERS

Why identify expertise?

- The Response Time Gap

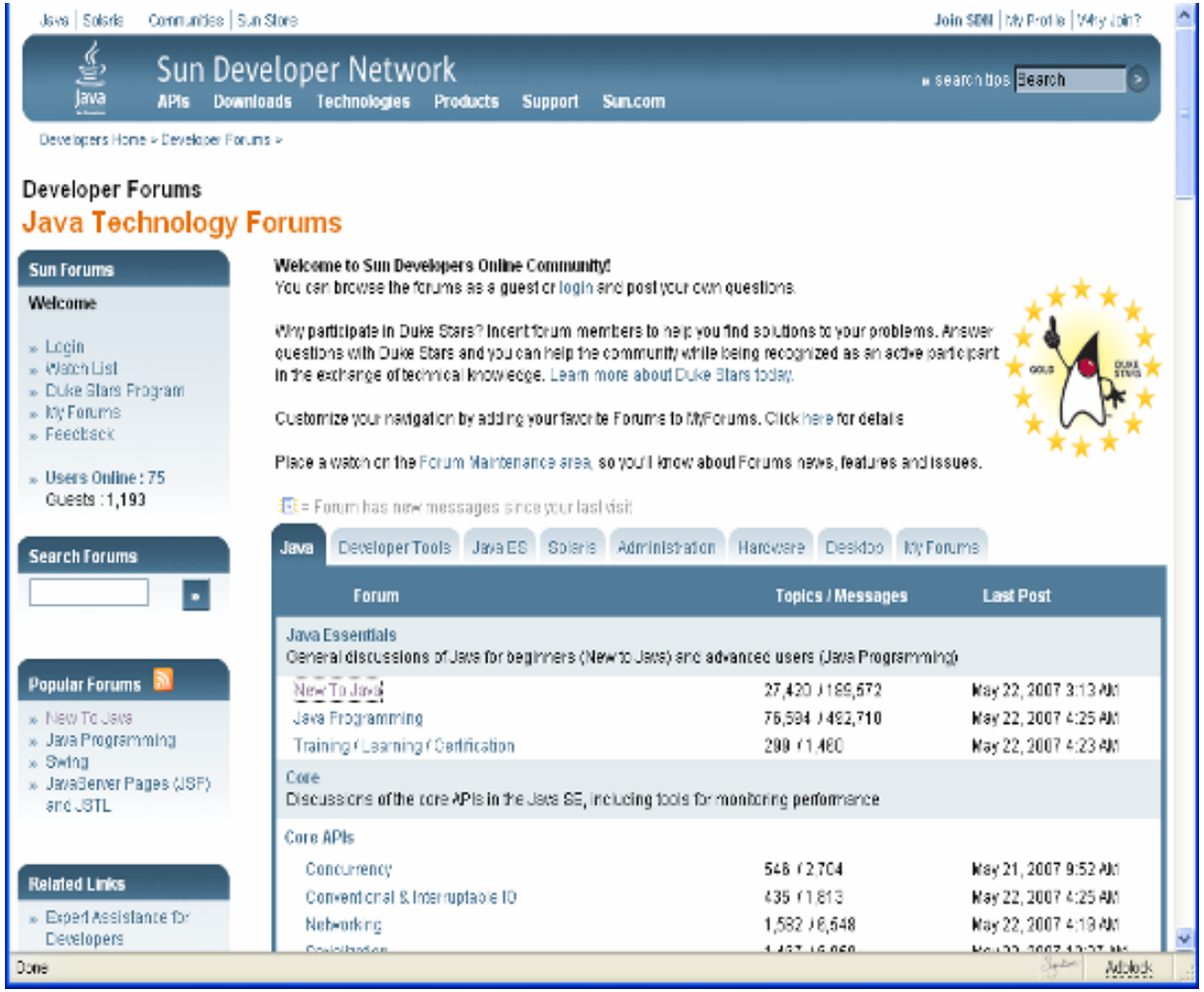


- The Expertise Gap
- Difficult to infer reliability of answers

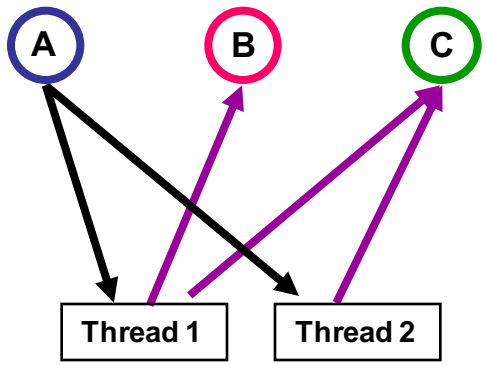
Automatically ranking expertise may be helpful.

Java Forum

- 87 sub-forums
- 1,438,053 messages
- community expertise network constructed:
  - 196,191 users
  - 796,270 edges

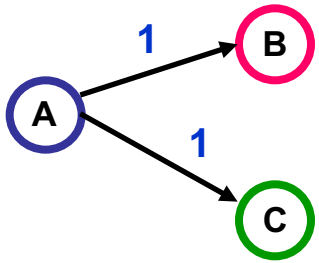


Constructing a community expertise network

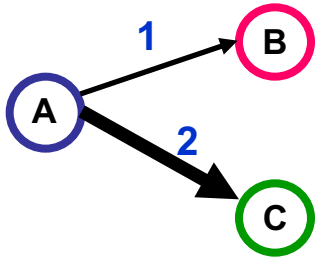


Thread 1*:* Large Data, binary search or hashtable? *user A*
Re: Large... *user B*
Re: Large... *user C*
Thread 2*:* Binary file with ASCII data *user A*
Re: File with... *user C*

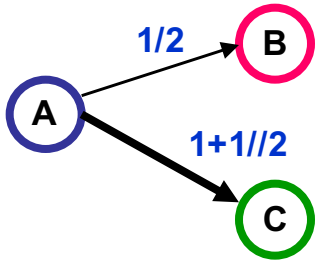
unweighted



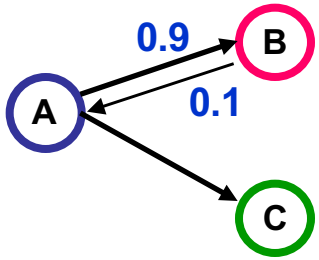
weighted by # threads



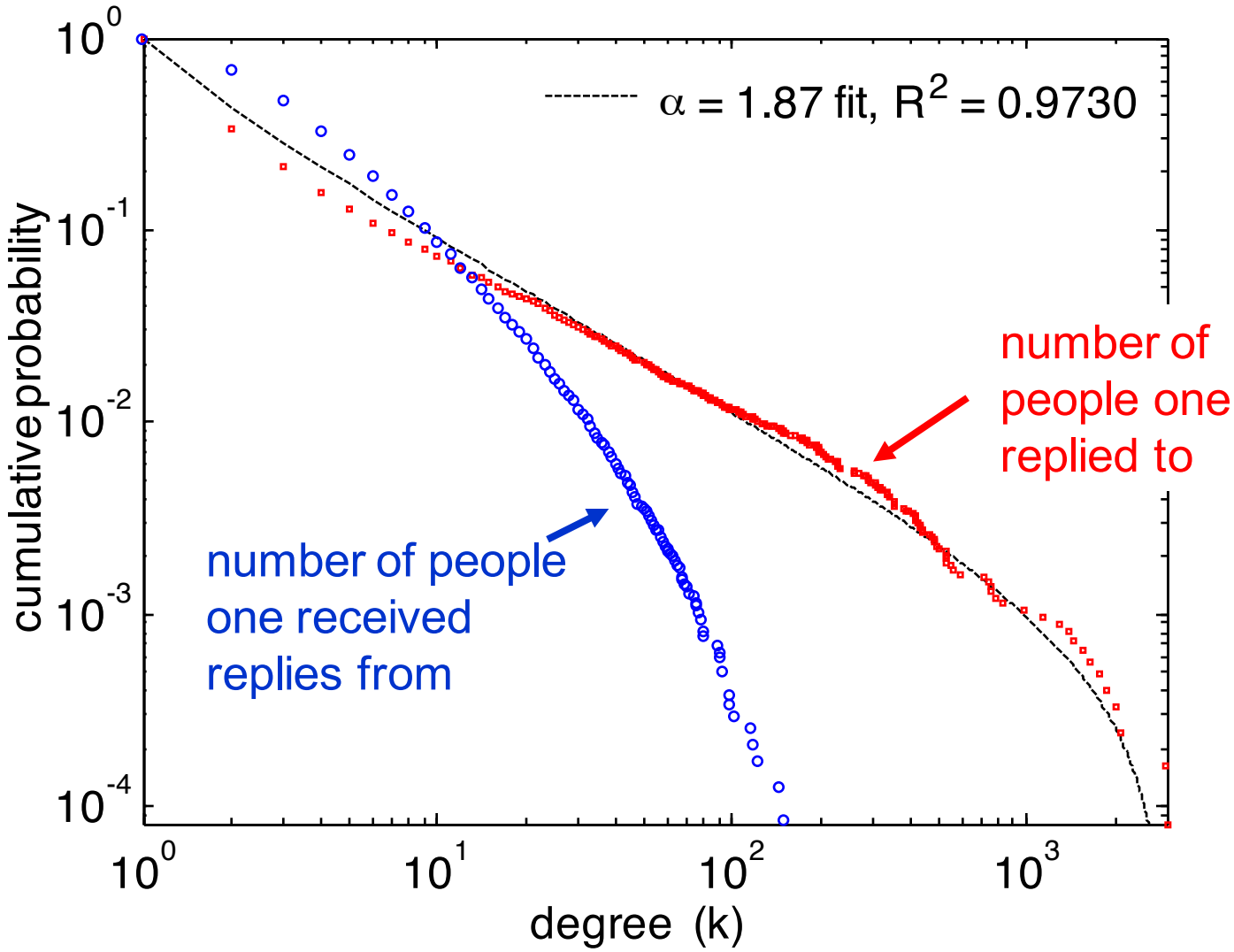
weighted by shared credit



weighted with backflow

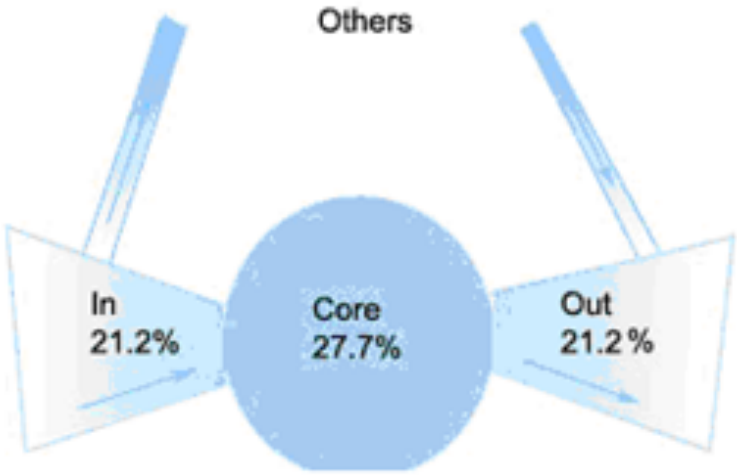


Uneven participation

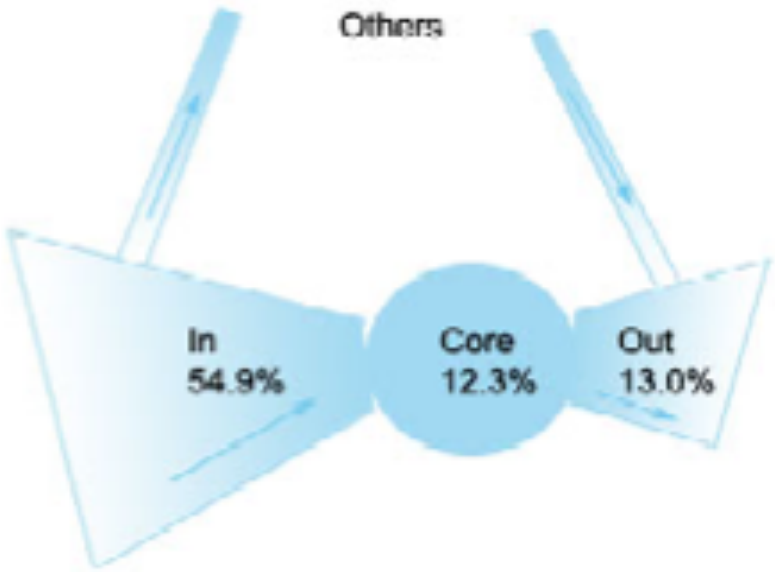


- 'answer people' may reply to thousands of others
- 'question people' are also uneven in the number of repliers to their posts, but to a lesser extent

Not Everyone Asks/Replies



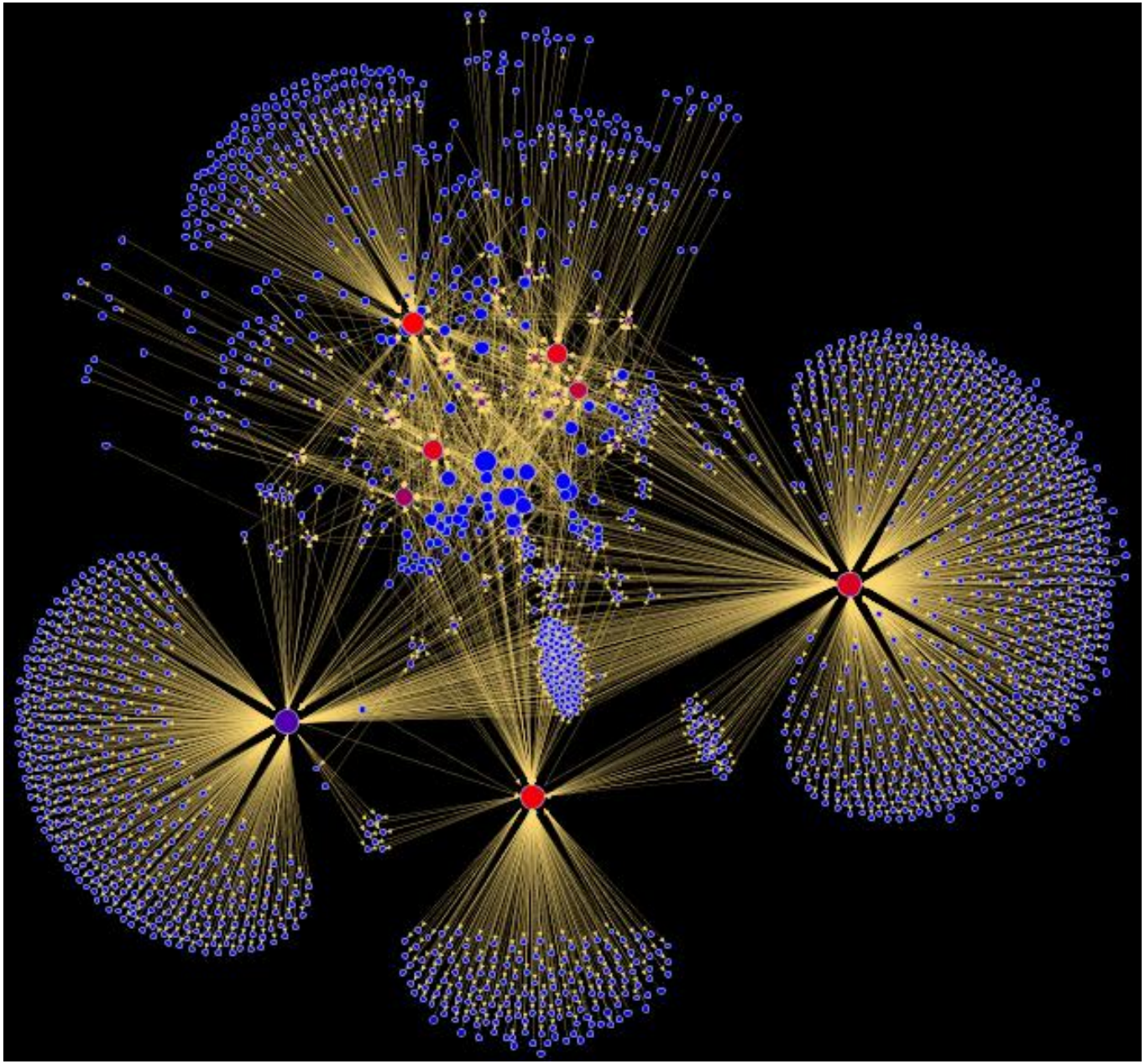
The Web is a bow tie



The Java Forum network is an uneven bow tie

- Core: A strongly connected component, in which everyone asks and answers
- IN: Mostly askers.
- OUT: Mostly Helpers

fragment of the Java Forum

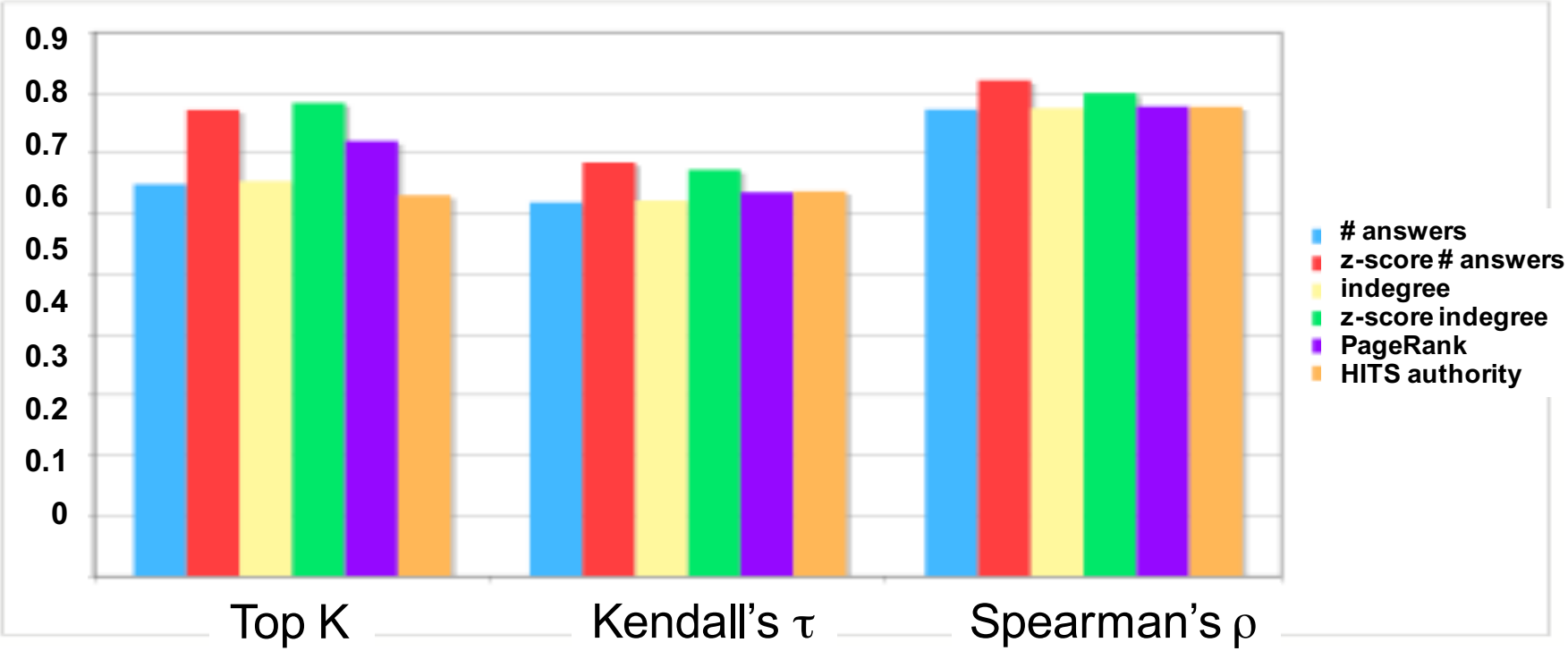


Relating network structure to Java expertise

- Human-rated expertise levels
  - 2 raters
  - 135 JavaForum users with >= 10 posts
  - inter-rater agreement ($\tau = 0.74$, $\rho = 0.83$)
  - for evaluation of algorithms, omit users where raters disagreed by more than 1 level ($\tau = 0.80$, $\rho = 0.83$)

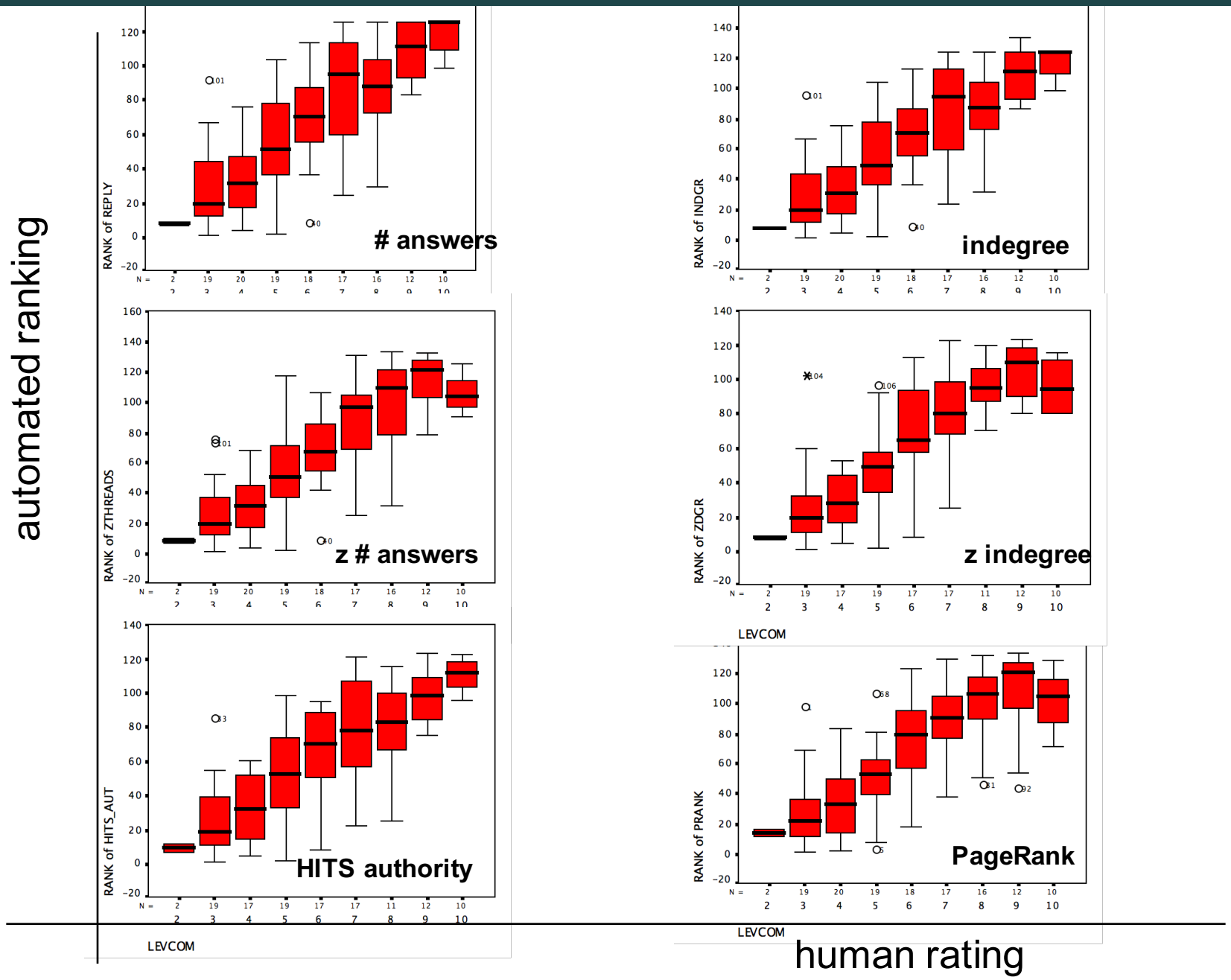
L	Category	Description
5	Top Java expert	Knows the core Java theory and related advanced topics deeply.
4	Java professional	Can answer all or most of Java concept questions. Also knows one or some sub topics very well,
3	Java user	Knows advanced Java concepts. Can program relatively well.
2	Java learner	Knows basic concepts and can program, but is not good at advanced topics of Java.
1	Newbie	Just starting to learn java.

Algorithm Rankings vs. Human Ratings

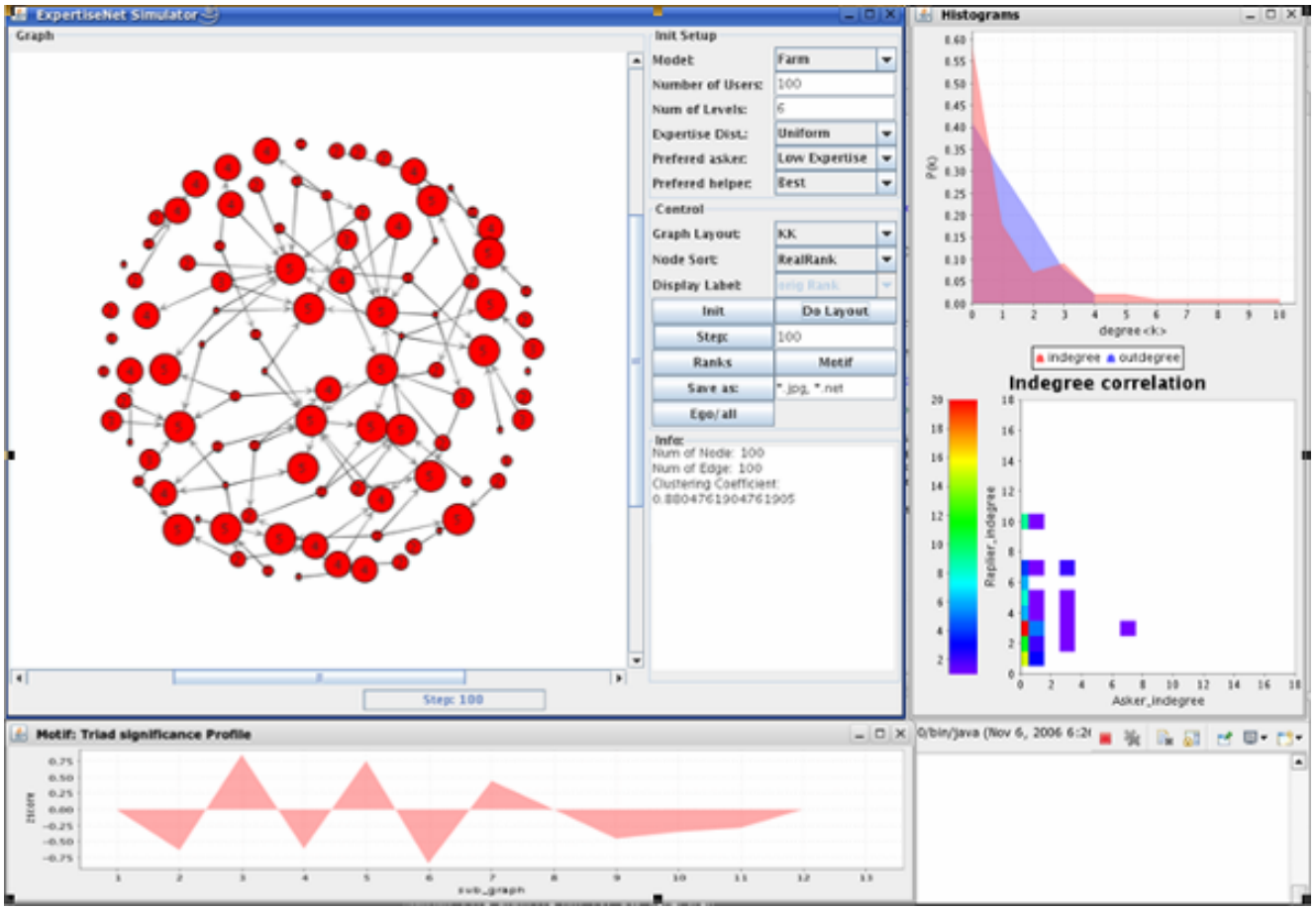


simple local measures do as well (and better) than measures incorporating the wider network topology

automated vs. human ratings



Modeling community structure to explain algorithm performance



ExpertiseNet Simulator

Control Parameters:

- Distribution of expertise
- Who asks questions most often?
- Who answers questions most often?
  - best expert most likely
  - someone a bit more expert

Simulating probability of expertise pairing

suppose:

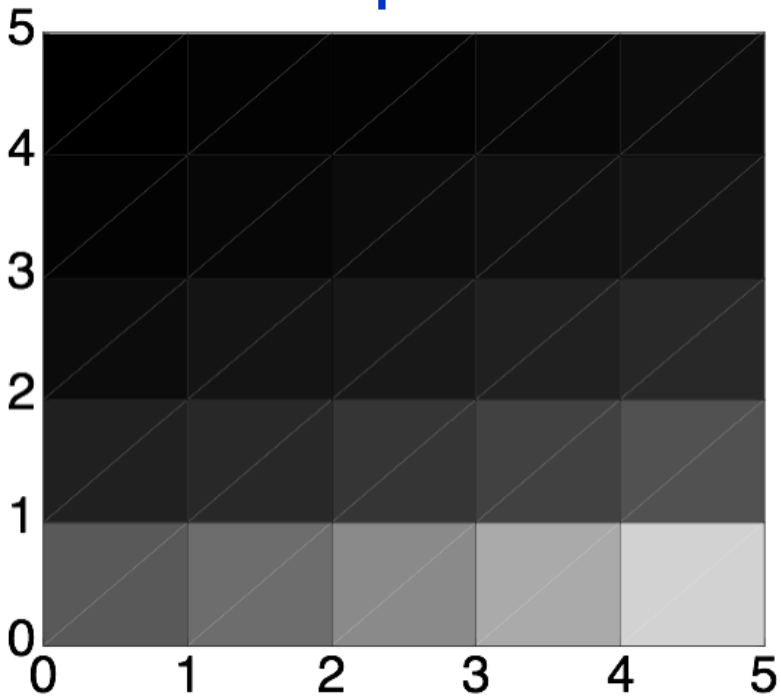
expertise is uniformly distributed

probability of posing a question is inversely proportional to expertise

p_{ij} = probability a user with expertise *j* replies to a user with expertise *i*

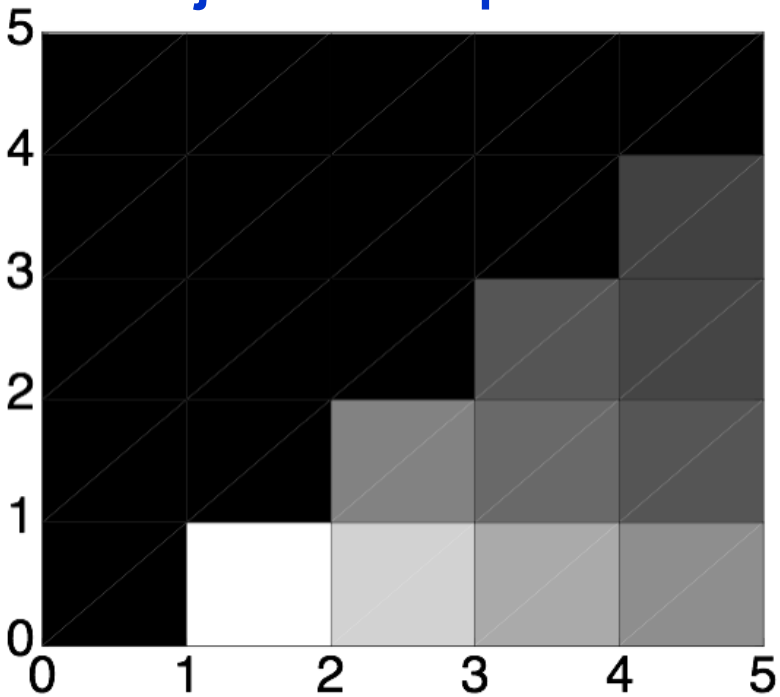
2 models:

'best' preferred



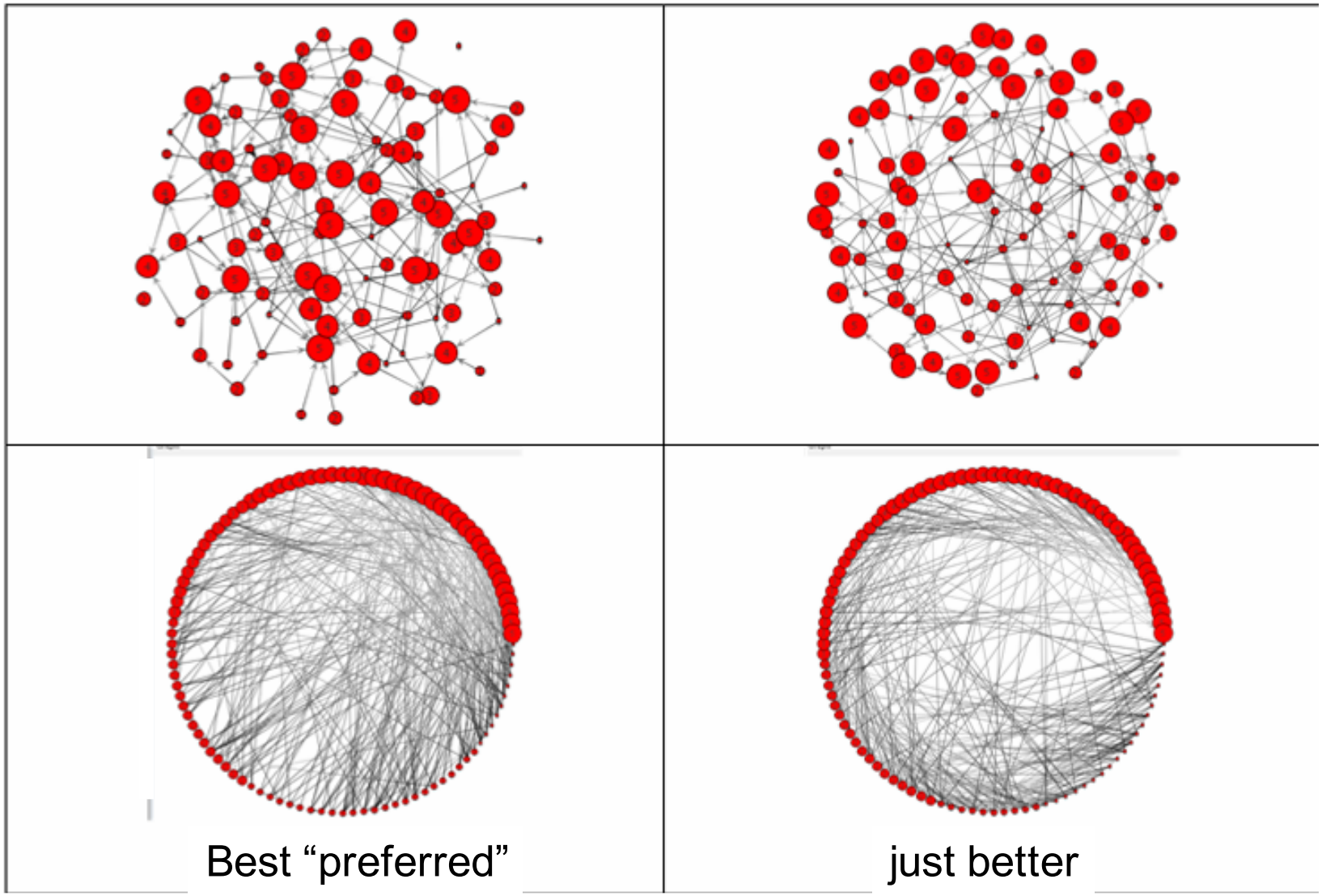
$$p_{ij} \sim e^{\beta(j-i)} / i$$

'just better' preferred

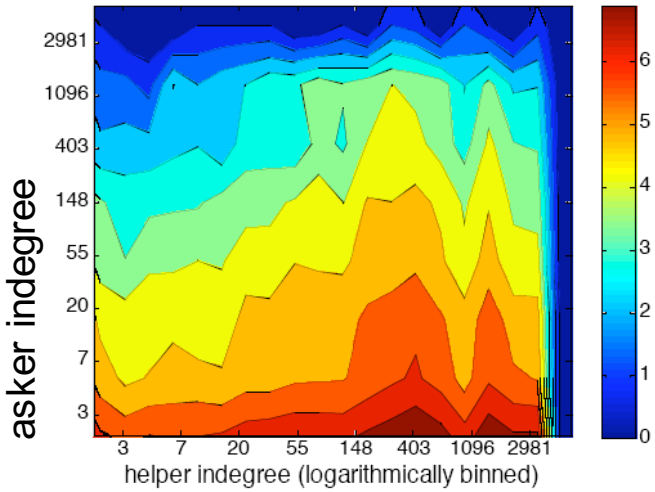


$$p_{ij} \sim e^{\gamma(i-j)} / i \qquad j>i$$

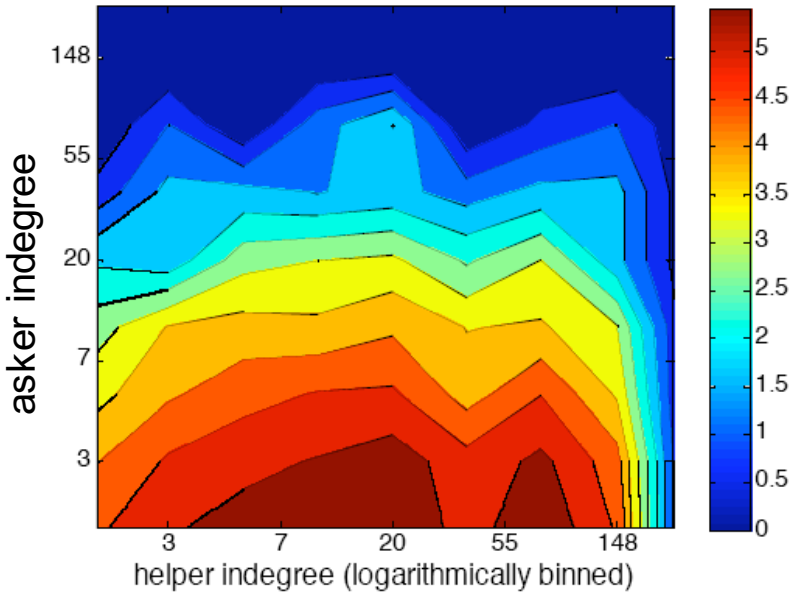
Visualization



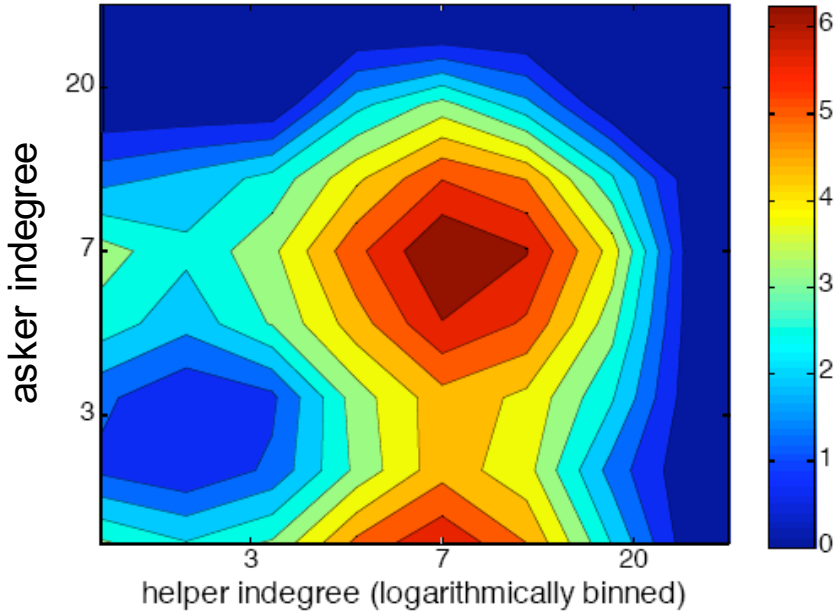
Degree correlation profiles



Java Forum Network

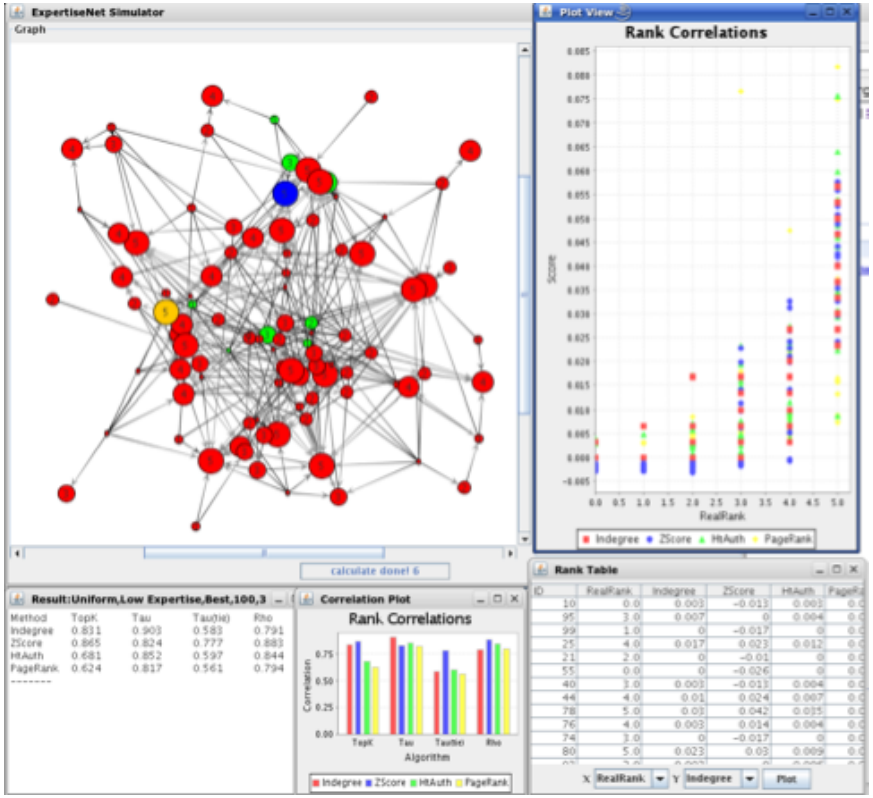


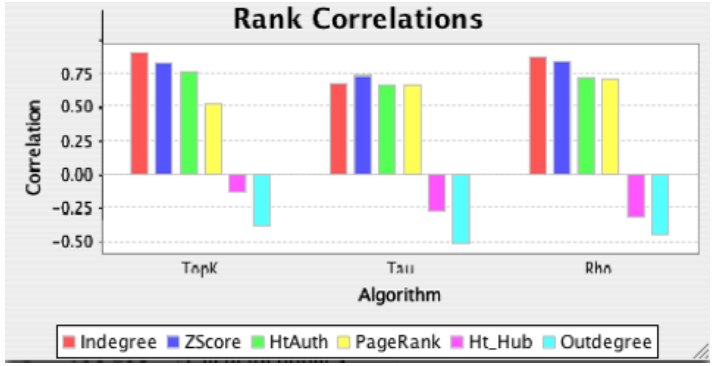
best preferred (simulation)



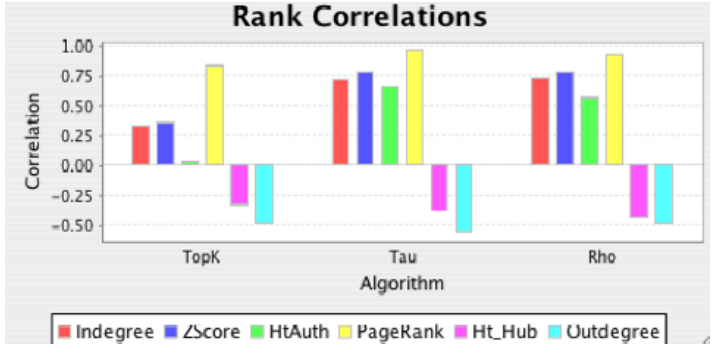
just better (simulation)

Simulation can tell us when to use which algorithms



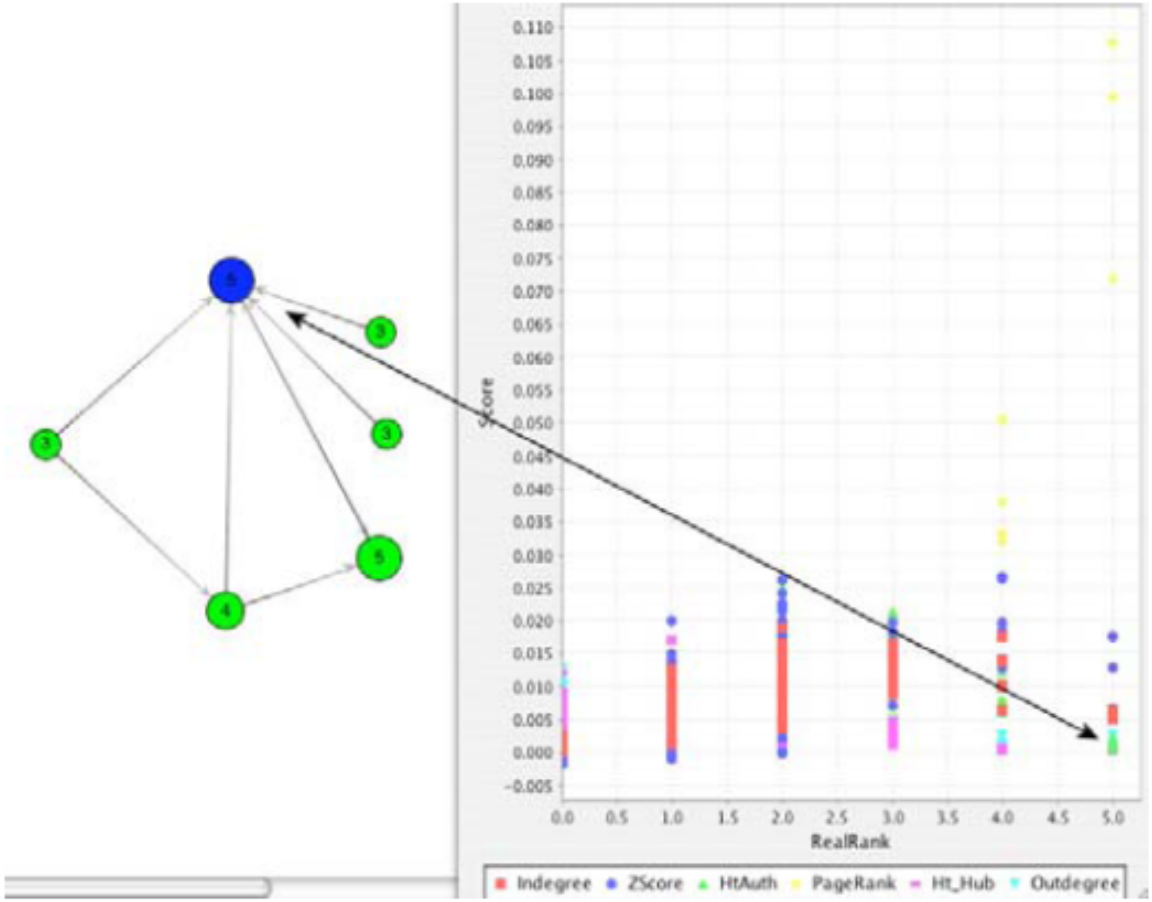


Preferred Helper: 'best available'



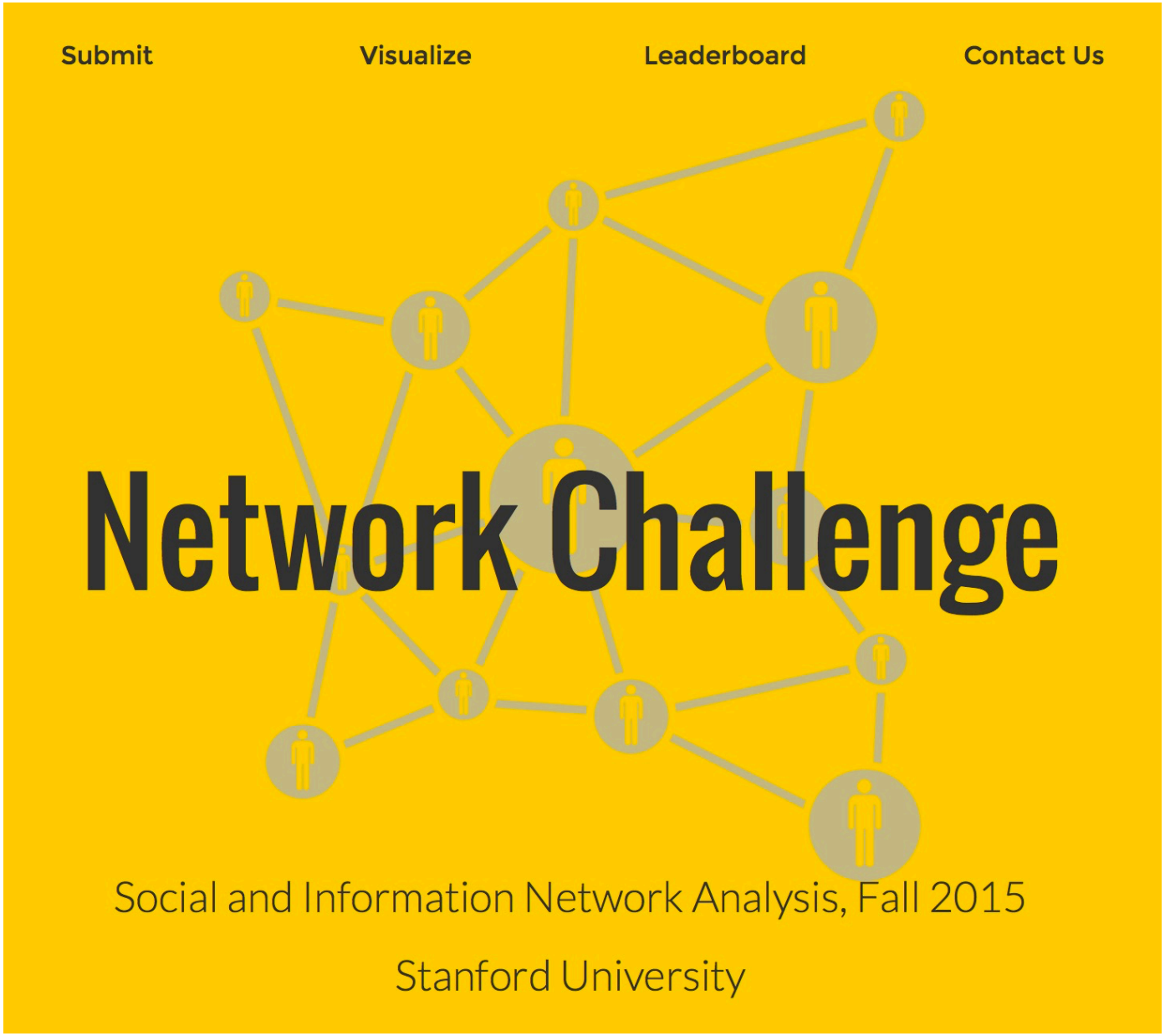
Preferred Helper: 'just better'

Different ranking algorithms perform differently



In the ‘just better’ model, a node is correctly ranked by PageRank but not by HITS. Can you explain why?

Coming soon…


Submit
Visualize
Leaderboard
Contact Us
Network Challenge
Social and Information Network Analysis, Fall 2015
Stanford University

SVD

Capturing network features

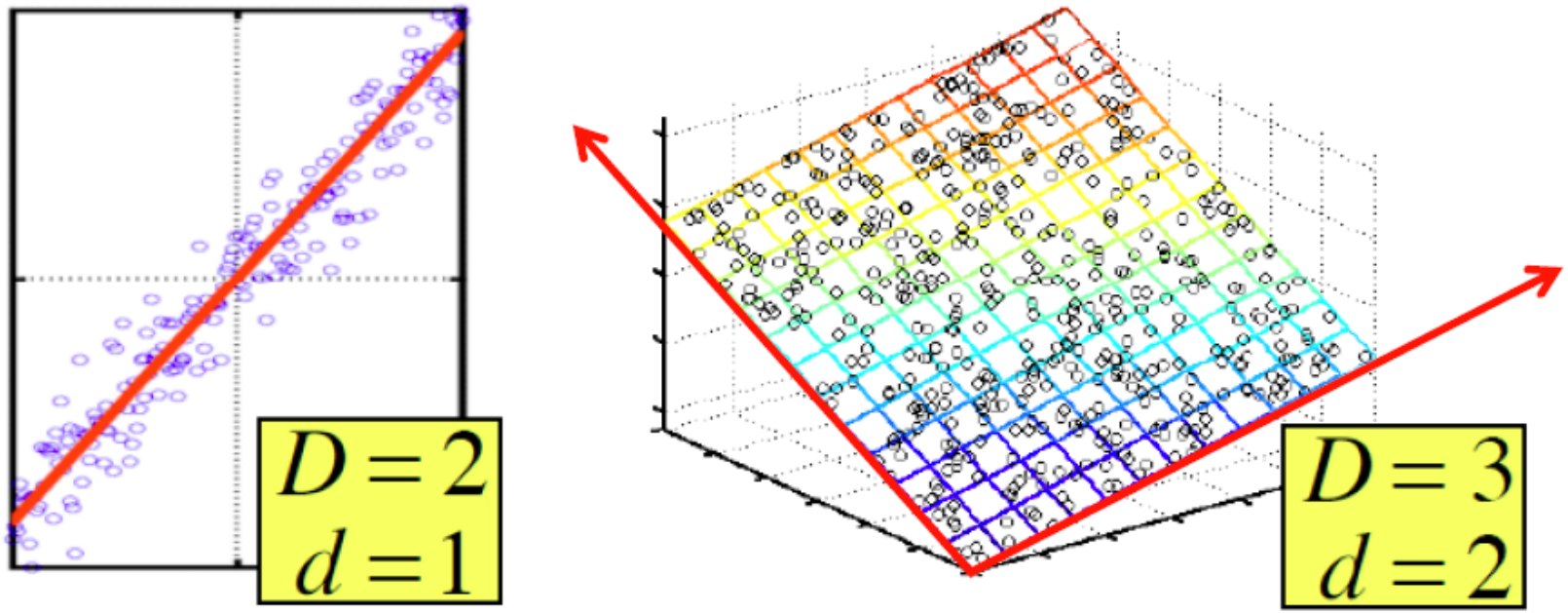
Dimensionality Reduction

- **Compress / reduce dimensionality:**
  - 10^6 rows; 10^3 columns; no updates
  - Random access to any cell(s); **small error: OK**

day / customer	We 7/10/96	Th 7/11/96	Fr 7/12/96	Sa 7/13/96	Su 7/14/96
ABC Inc.	1	1	1	0	0
DEF Ltd.	2	2	2	0	0
GHI Inc.	1	1	1	0	0
KLM Co.	5	5	5	0	0
Smith	0	0	0	2	2
Johnson	0	0	0	3	3
Thompson	0	0	0	1	1

The above matrix is really "2-dimensional." All rows can be reconstructed by scaling [1 1 1 0 0] or [0 0 0 1 1]

Dimensionality Reduction



- **Assumption:** Data lies on or near a low *d*-dimensional subspace
- **Axes of this subspace are effective representation of the data**

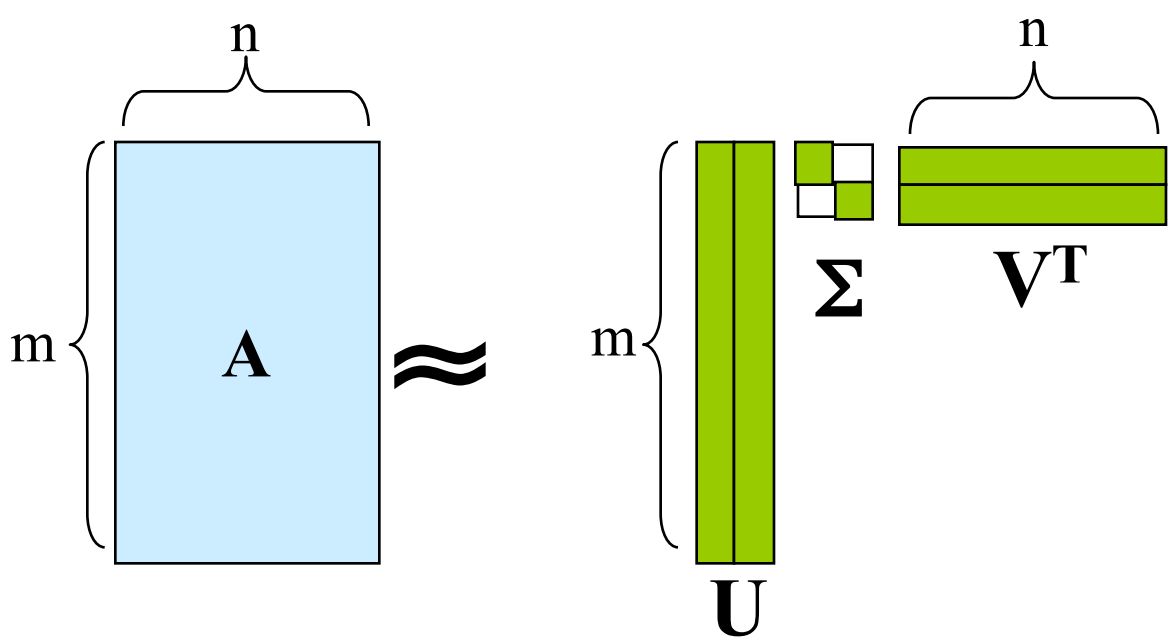
SVD - Definition

$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

- **A**:
  - *m* x *n* matrix (e.g., *m* documents, *n* terms)
- **U**:
  - *m* x *r* matrix (*m* documents, *r* concepts)
- **Σ**:
  - *r* x *r* diagonal matrix (strength of each 'concept') (*r* : rank of the matrix **A**)
- **V**:
  - *n* x *r* matrix (*n* terms, *r* concepts)

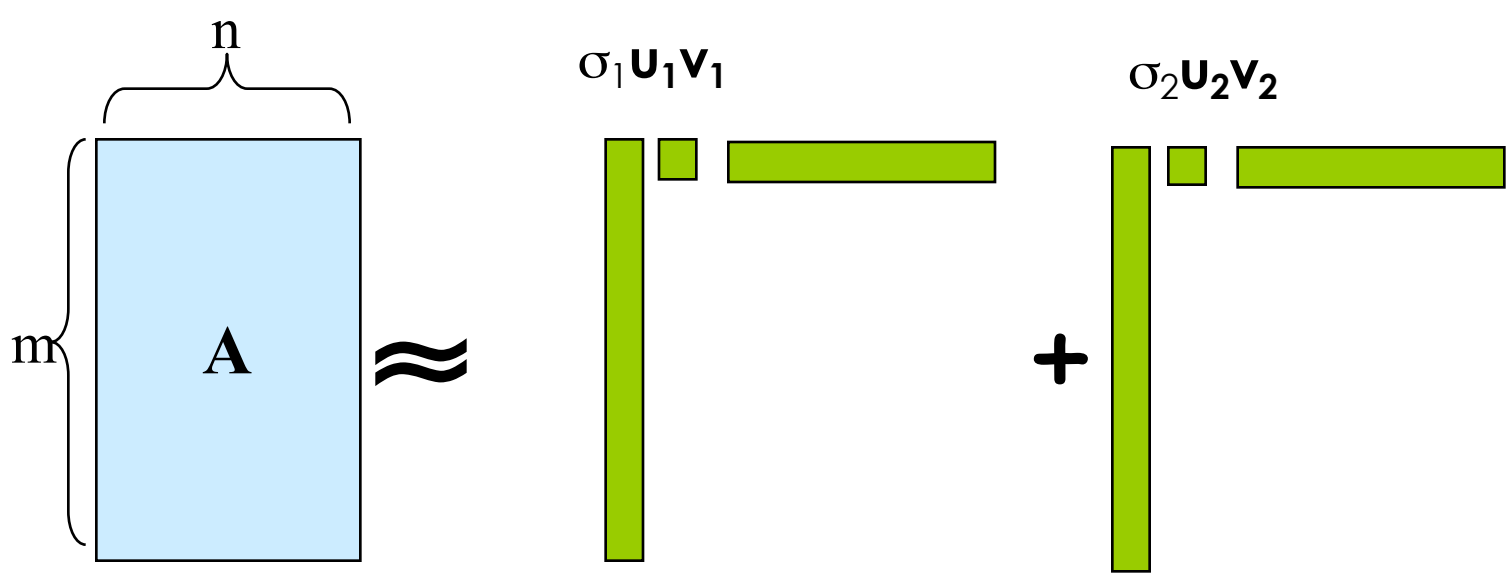
SVD

$$\mathbf{A} \approx \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^T$$



SVD

$$\mathbf{A} \approx \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^T$$



σ_i … scalar

u_i … vector

v_i … vector

SVD - Properties

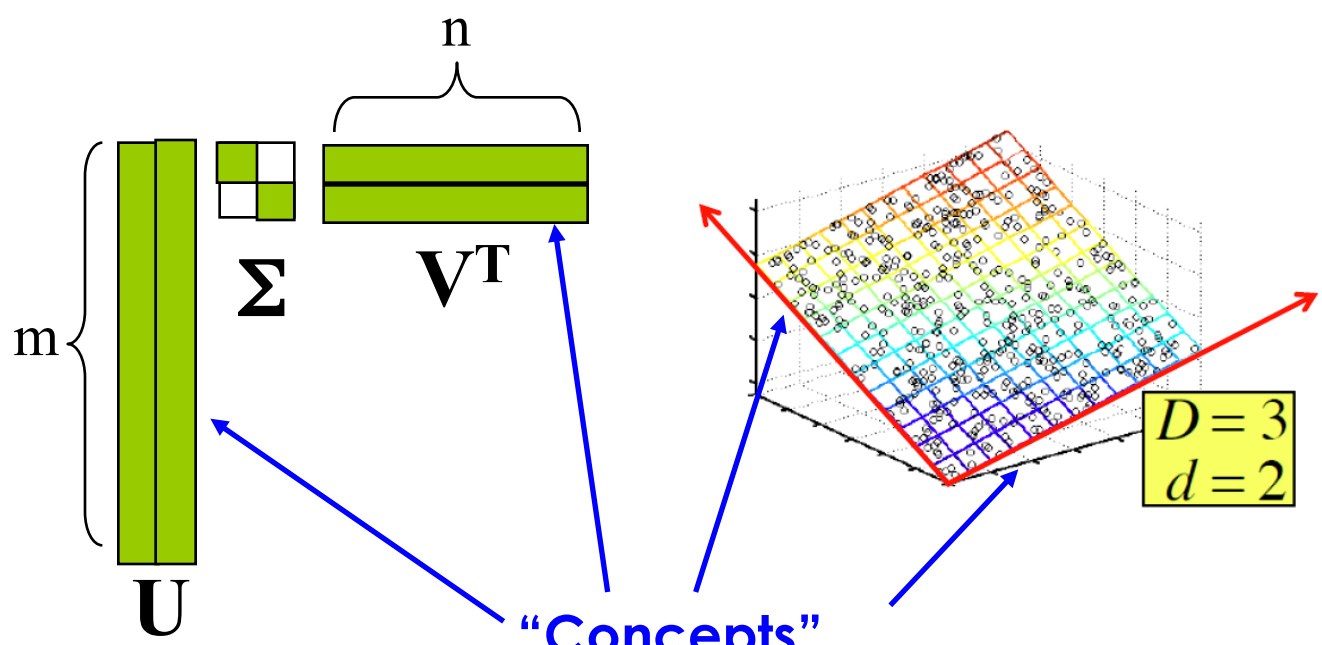
It is **always** possible to decompose a real matrix $\boldsymbol{A}$ into $\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{\Sigma}\,\boldsymbol{V}^T$, where

- $\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}$: unique
- $\boldsymbol{U}, \boldsymbol{V}$: column orthonormal
  - $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$; $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}$ ($\boldsymbol{I}$: identity matrix)
  - (Columns are orthogonal unit vectors)
- $\boldsymbol{\Sigma}$: diagonal
  - Entries (**singular values**) are positive, and sorted in decreasing order ($\sigma_1 \geq \sigma_2 \geq \ldots \geq 0$)

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example: Users to Movies

$$
\begin{array}{c|c}
 & \begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{array}{c} \text{SciFi} \\ \\ \\ \\ \text{Romance} \\ \\ \\ \end{array} &
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
3 & 3 & 3 & 0 & 0 \\
4 & 4 & 4 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 2 & 0 & 4 & 4 \\
0 & 0 & 0 & 5 & 5 \\
0 & 1 & 0 & 2 & 2
\end{bmatrix}
\end{array}
=
$$



"Concepts"
AKA Latent dimensions
AKA Latent factors

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example: Users to Movies

SciFi ↑ ↓ Romance ↑ ↓

$$
\begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
3 & 3 & 3 & 0 & 0 \\
4 & 4 & 4 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 2 & 0 & 4 & 4 \\
0 & 0 & 0 & 5 & 5 \\
0 & 1 & 0 & 2 & 2
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
0.13 & 0.02 & -0.01 \\
0.41 & 0.07 & -0.03 \\
0.55 & 0.09 & -0.04 \\
0.68 & 0.11 & -0.05 \\
0.15 & -0.59 & 0.65 \\
0.07 & -0.73 & -0.67 \\
0.07 & -0.29 & 0.32
\end{bmatrix}
\times
\begin{bmatrix}
12.4 & 0 & 0 \\
0 & 9.5 & 0 \\
0 & 0 & 1.3
\end{bmatrix}
\times
\begin{bmatrix}
0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\
0.40 & -0.80 & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example: Users to Movies

U is "user-to-concept" similarity matrix

SciFi-concept

Romance-concept

$$
\begin{array}{c}
\text{SciFi} \uparrow \\ \downarrow \\ \\ \\ \\ \text{Romance} \uparrow \\ \downarrow
\end{array}
\begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
3 & 3 & 3 & 0 & 0 \\
4 & 4 & 4 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 2 & 0 & 4 & 4 \\
0 & 0 & 0 & 5 & 5 \\
0 & 1 & 0 & 2 & 2
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
0.13 & 0.02 & -0.01 \\
0.41 & 0.07 & -0.03 \\
0.55 & 0.09 & -0.04 \\
0.68 & 0.11 & -0.05 \\
0.15 & -0.59 & 0.65 \\
0.07 & -0.73 & -0.67 \\
0.07 & -0.29 & 0.32
\end{bmatrix}
\times
\begin{bmatrix}
12.4 & 0 & 0 \\
0 & 9.5 & 0 \\
0 & 0 & 1.3
\end{bmatrix}
\times
\begin{bmatrix}
0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\
0.40 & -0.80 & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example: Users to Movies

SciFi-concept

"strength" of the SciFi-concept

	Matrix	Alien	Serenity	Casablanca	Amelie
SciFi ↑	1	1	1	0	0
	3	3	3	0	0
	4	4	4	0	0
↓	5	5	5	0	0
Romance ↑	0	2	0	4	4
	0	0	0	5	5
↓	0	1	0	2	2

$$= \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example:

$$
\begin{array}{c}
\text{SciFi} \\
\text{Romance}
\end{array}
\begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
3 & 3 & 3 & 0 & 0 \\
4 & 4 & 4 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 2 & 0 & 4 & 4 \\
0 & 0 & 0 & 5 & 5 \\
0 & 1 & 0 & 2 & 2
\end{bmatrix}
\end{array}
=
\begin{array}{c}
\begin{array}{ccc} \text{SciFi-concept} & & \end{array} \\
\begin{bmatrix}
0.13 & 0.02 & -0.01 \\
0.41 & 0.07 & -0.03 \\
0.55 & 0.09 & -0.04 \\
0.68 & 0.11 & -0.05 \\
0.15 & -0.59 & 0.65 \\
0.07 & -0.73 & -0.67 \\
0.07 & -0.29 & 0.32
\end{bmatrix}
\end{array}
\times
\begin{bmatrix}
12.4 & 0 & 0 \\
0 & 9.5 & 0 \\
0 & 0 & 1.3
\end{bmatrix}
\times
\begin{array}{c}
\begin{array}{ccc} \text{SciFi-concept} & & \end{array} \\
\begin{bmatrix}
0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\
0.40 & -0.80 & 0.40 & 0.09 & 0.09
\end{bmatrix}
\end{array}
$$

V **is "movie-to-concept" similarity matrix**

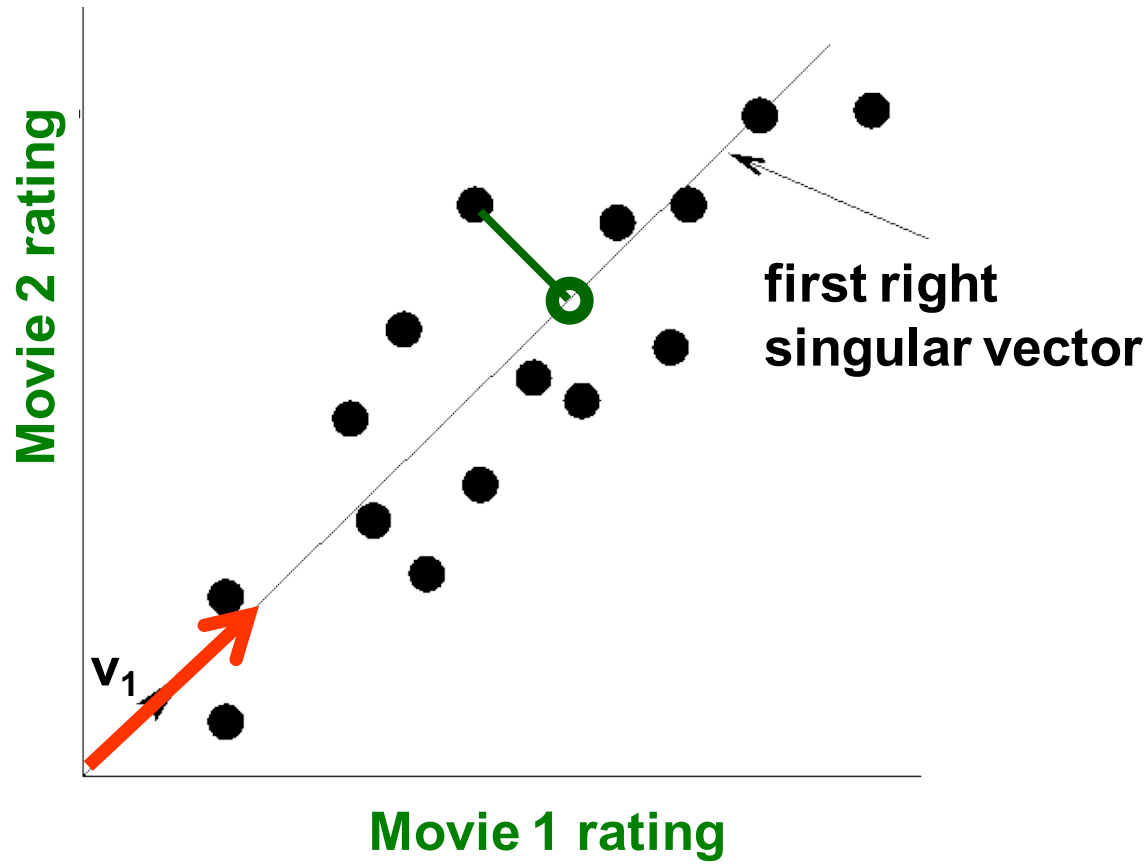
SVD - Interpretation #1

‘**movies**’, ‘**users**’ and ‘**concepts**’:

- ***U***: user-to-concept similarity matrix
- ***V***: movie-to-concept similarity matrix
- **Σ**: its diagonal elements:
  ‘strength’ of each concept

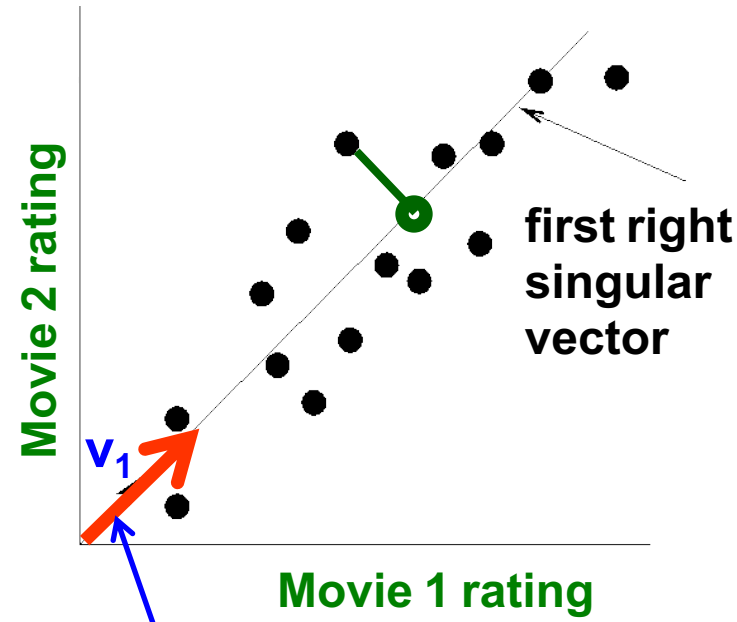
SVD - interpretation #2

- **SVD gives 'best' axis to project on:**
  - '**best**' = min sum of squares of projection errors
- **In other words, minimum reconstruction error**



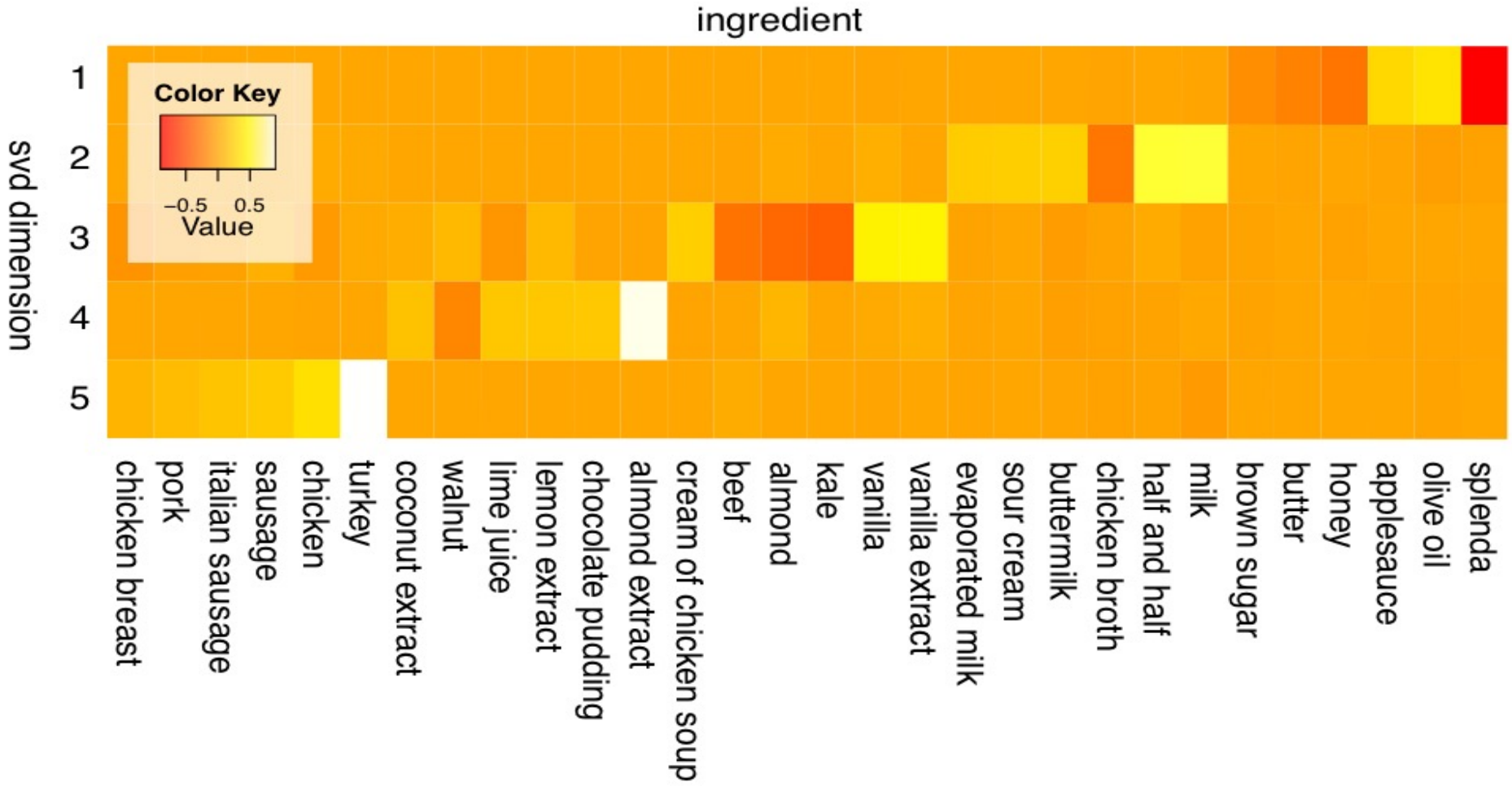
SVD - Interpretation #2

- **$A = U \Sigma V^T$ - example:**
  - **V**: "movie-to-concept" matrix
  - **U**: "user-to-concept" matrix



$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

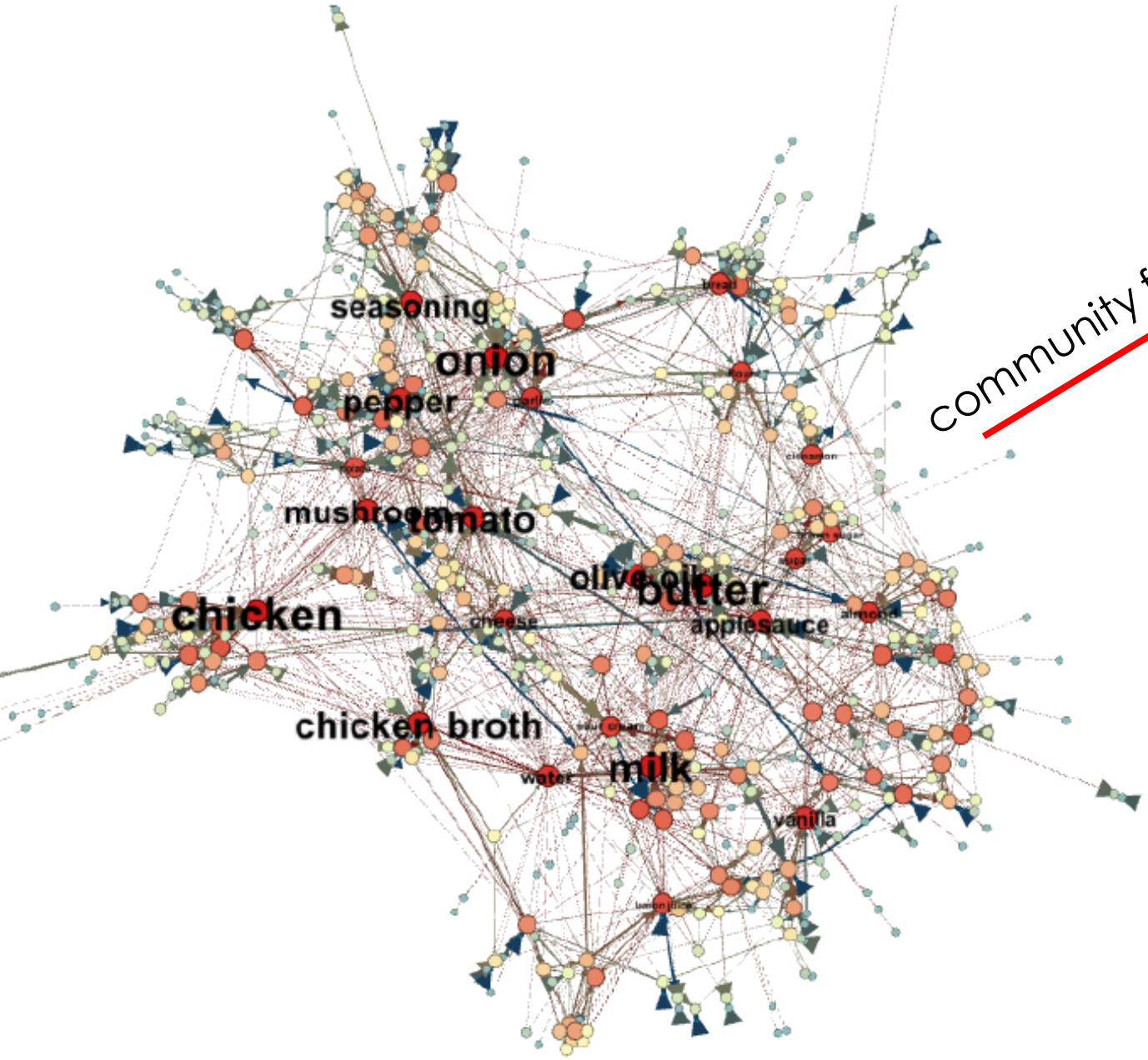
SVD – ingredient networks



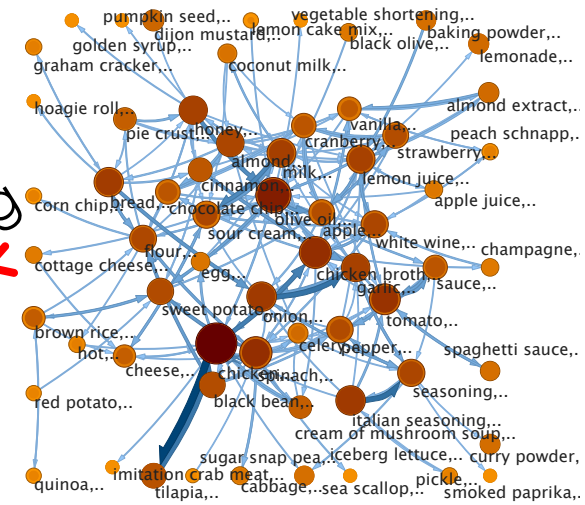
Substitution network

- Extract substitution relationships from comments
  - e.g. "I replaced the butter with sour cream"
  - "replace **a** with **b**", "substitute **b** for **a**", "**b** instead of **a**"
- Nodes: ingredients
- Edge weights = p(**b** | **a**), which is the proportion of substitutions of ingredient **a** that suggests ingredient **b**

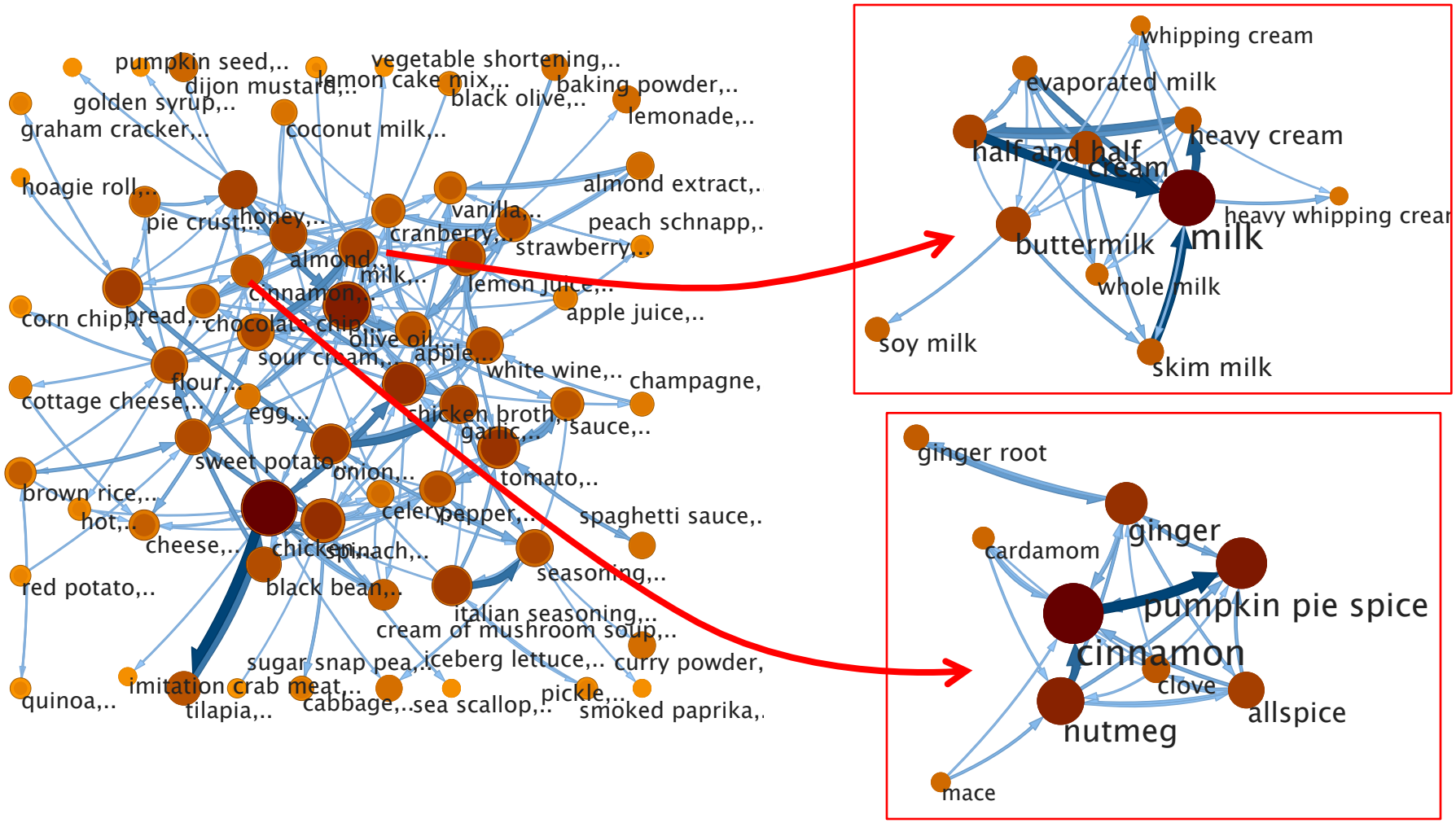
Substitution network



community finding

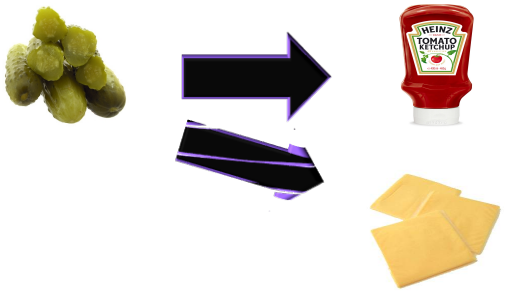


Substitution network: communities



Substitution network and users' preference

- Preference network
  - Create an edge from ingredient **a** to **b** if rating(**a**) < rating(**b**)
  - ex:
    - Recipe X contains 
    - Recipe Y contains 
    - Rating(X) > Rating(Y)



Substitute network and users' preference

- Weight of preference network
  - PMI(a->b)=log(p(a->b)/p(a)p(b))
  - where p(a->b)= (# of recipe pairs from a to b)/(# of recipe pairs)
- Correlations between preference network and substitute network (ρ = 0.72, p<0.001)

Prediction task

- Given a recipe pair with overlapped ingredients, determine which one has the higher rating

Prediction task

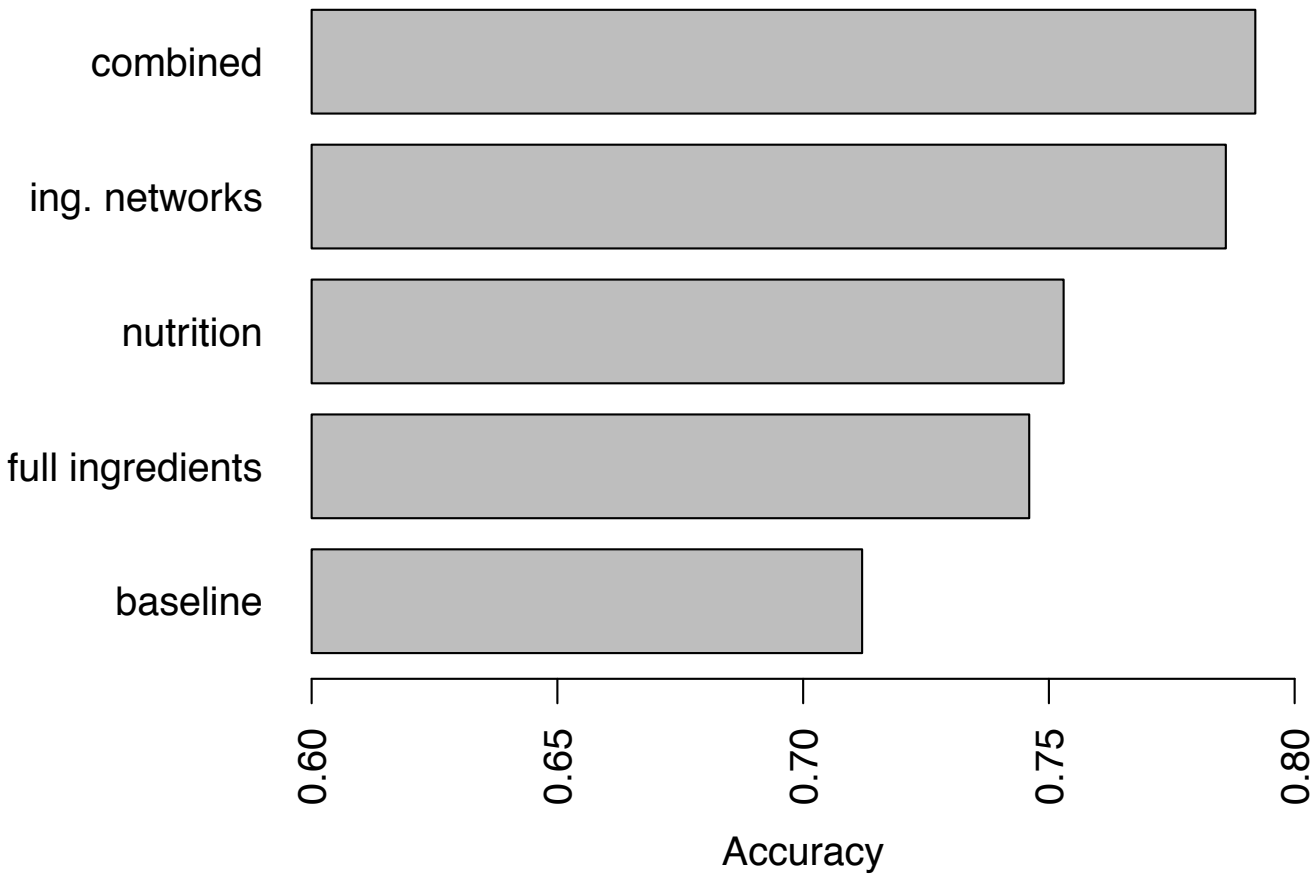
- Features
  - Baseline
    - Cooking methods, preparation time, the number of servings
  - 1000 popular ingredient list
    - Binary vector indicating the occurrence of ingredients
  - Nutrition
    - Calories, carbohydrates, fat, etc.
  - Ingredient networks
    - Network positions (centrality) and communities (SVD)
  - Combined set
    - Everything listed above

Prediction task

- 62,031 recipe pairs (X,Y)
  - where rating(X) > rating(Y)
  - >= 10 user reviews
  - >= 50% users have rated both recipes
  - Cosine similarity of ingredients (X,Y)> 0.2
- Train with gradient boosting tree
  - balanced dataset
  - 2/3 for training, 1/3 for testing
  - Evaluate based on accuracy

Prediction performance

- Ingredient network features lead to improved performance



Summary: SNA applications II

- Network structure can be used to rank nodes
  - HITS
  - PageRank
- SVD can represent network features compactly
- The application possibilities are endless