Graphs / Networks

Centrality measures, algorithms, Interactive applications

Duen Horng (Polo) Chau

Associate Professor, College of Computing
Associate Director, MS Analytics
Georgia Tech

Mahdi Roozbahani

Lecturer, Computational Science & Engineering, Georgia Tech
Founder of Filio, a visual asset management platform

Partly based on materials by
Professors Guy Lebanon, Jeffrey Heer, John Stasko, Christos Faloutsos

Centrality

= “Importance”

Why Node Centrality?

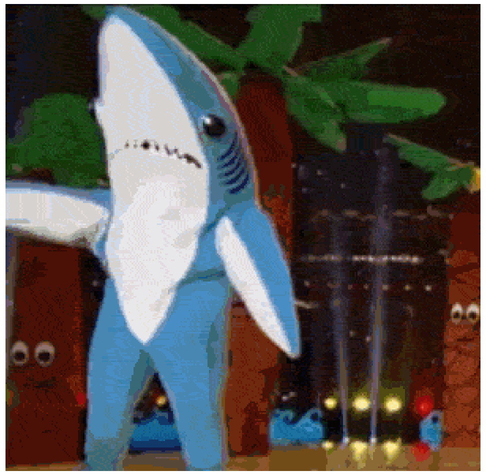
What can we do if we can rank all the nodes in a graph (e.g., Facebook, LinkedIn, Twitter)?

Why Node Centrality?

What can we do if we can rank all the nodes in a graph (e.g., Facebook, LinkedIn, Twitter)?

- Find **celebrities** or influential people in a social network (Twitter)
- Find "**gatekeepers**" who connect communities (headhunters love to find them on LinkedIn)
- What else?





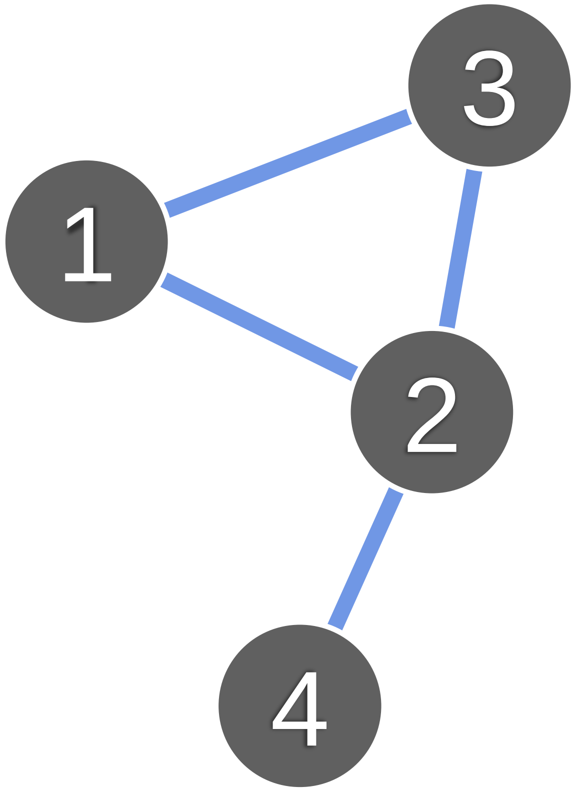
Why Node Centrality?

Helps **graph analysis, visualization, understanding**, e.g.,

- Let us **rank** nodes, group or study them by centrality
- Only show subgraph formed by the **top 100 nodes**, out of the millions in the full graph
    - **Similar to google search results** (ranked, and they only show you 10 per page)
- Most graph analysis packages already have centrality algorithms implemented. **Use them!**

Can also compute edge centrality.
Here we focus on node centrality.

Degree Centrality (easiest)



Degree = number of neighbors

- For directed graphs
  - **In degree** = No. of incoming edges
  - **Out degree** = No. of outgoing edges
- For undirected graphs, **only degree is defined**.
- Algorithms?
  - Sequential scan through edge list
  - What about for a graph stored in SQLite?

1, 2
1, 3
2, 4
3, 2

Computing Degrees using SQL

Recall simplest way to store a graph in SQLite:

```
edges(source_id, target_id)
```

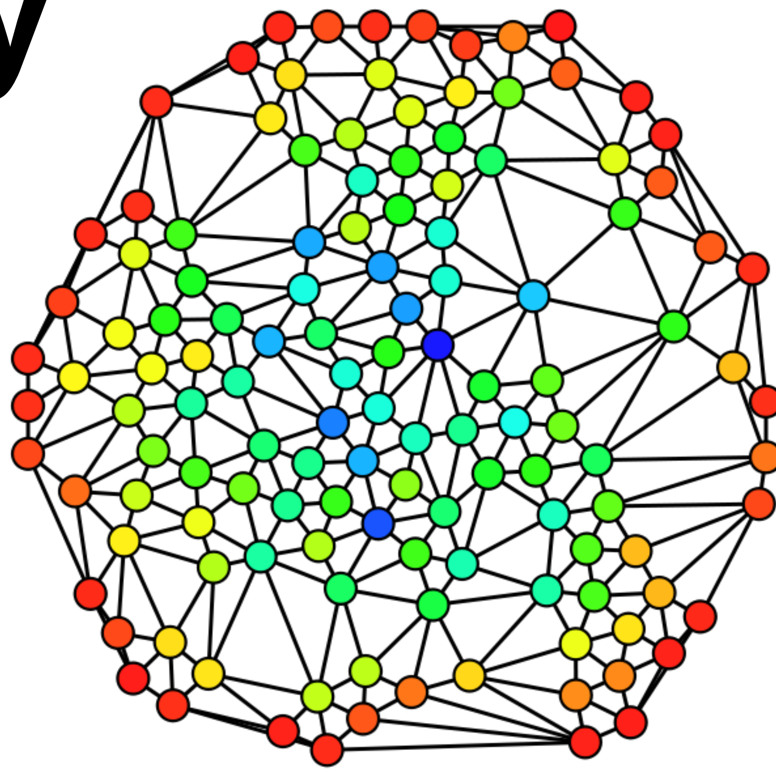
1, 2
1, 3
2, 4
3, 2

1. If slow, first create index for each column

2. Use **group by** statement to find **out degrees**

```
select count(*) from edges group by source_id;
```

Betweenness Centrality



High betweenness = “gatekeeper”

Betweenness of a node v

$$= \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

Number of shortest paths between s and t that **goes through v**

Number of shortest paths between s and t

= how often a node serves as the “bridge” that connects two other nodes.

Betweenness is very well studied. http://en.wikipedia.org/wiki/Centrality#Betweenness_centrality

(Local) Clustering Coefficient

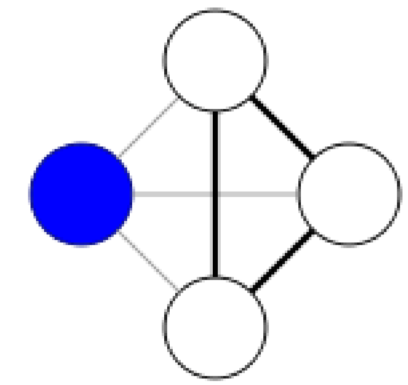
A node's clustering coefficient is a measure of how close the node's neighbors are from forming a clique.

1 = neighbors form a clique

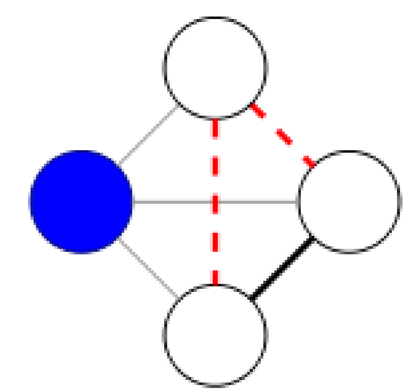
0 = No edges among neighbors

(Assuming undirected graph)

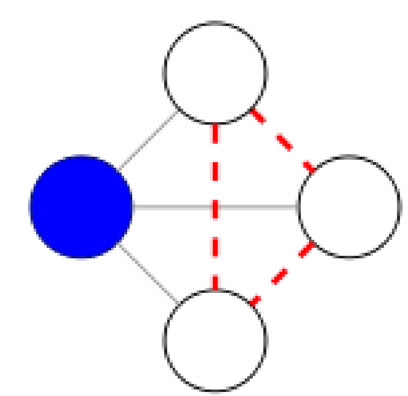
"Local" means it's for a node; can also compute a graph's "global" coefficient



c = 1



c = 1/3



c = 0

Image source: http://en.wikipedia.org/wiki/Clustering_coefficient

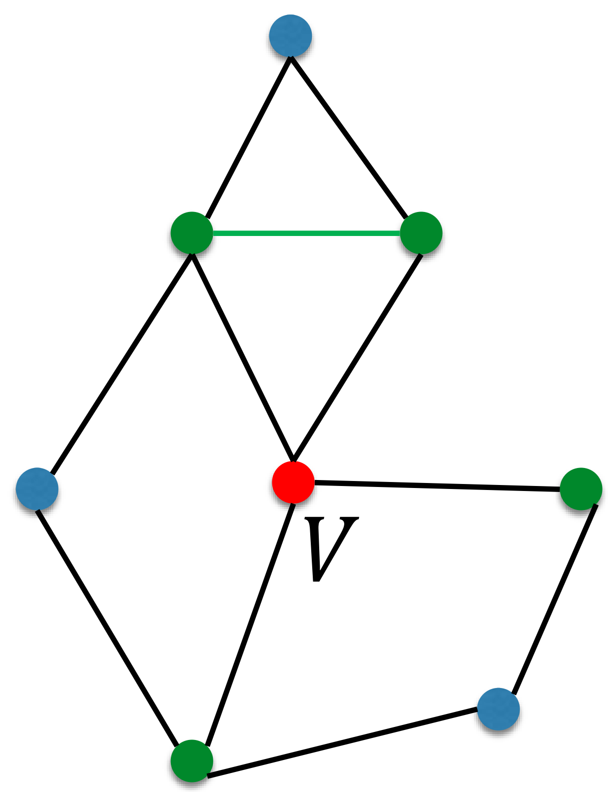
(Local) Clustering Coefficient

V: a node

$\boldsymbol{K_V}$: Number of edges

$\boldsymbol{N_V}$: Number of links between neighbors of V

$$CC(V) = \frac{N_V}{\frac{K_V(K_V - 1)}{2}}$$



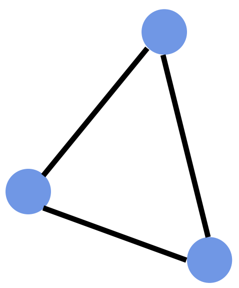
$\boldsymbol{N_V = 1}$
$\boldsymbol{K_V = 4}$

Computing Clustering Coefficients...

Requires **triangle counting**

Real social networks have a lot of triangles

- Friends of friends are friends



Triangles are **expensive** to compute

(neighborhood intersections; several approx. algos)

Can we do that quickly?

Algorithm details:
Faster Clustering Coefficient Using Vertex Covers
http://www.cc.gatech.edu/~ogreen3/_docs/2013VertexCoverClusteringCoefficients.pdf

details

Super Fast Triangle Counting [Tsourakakis ICDM 2008]

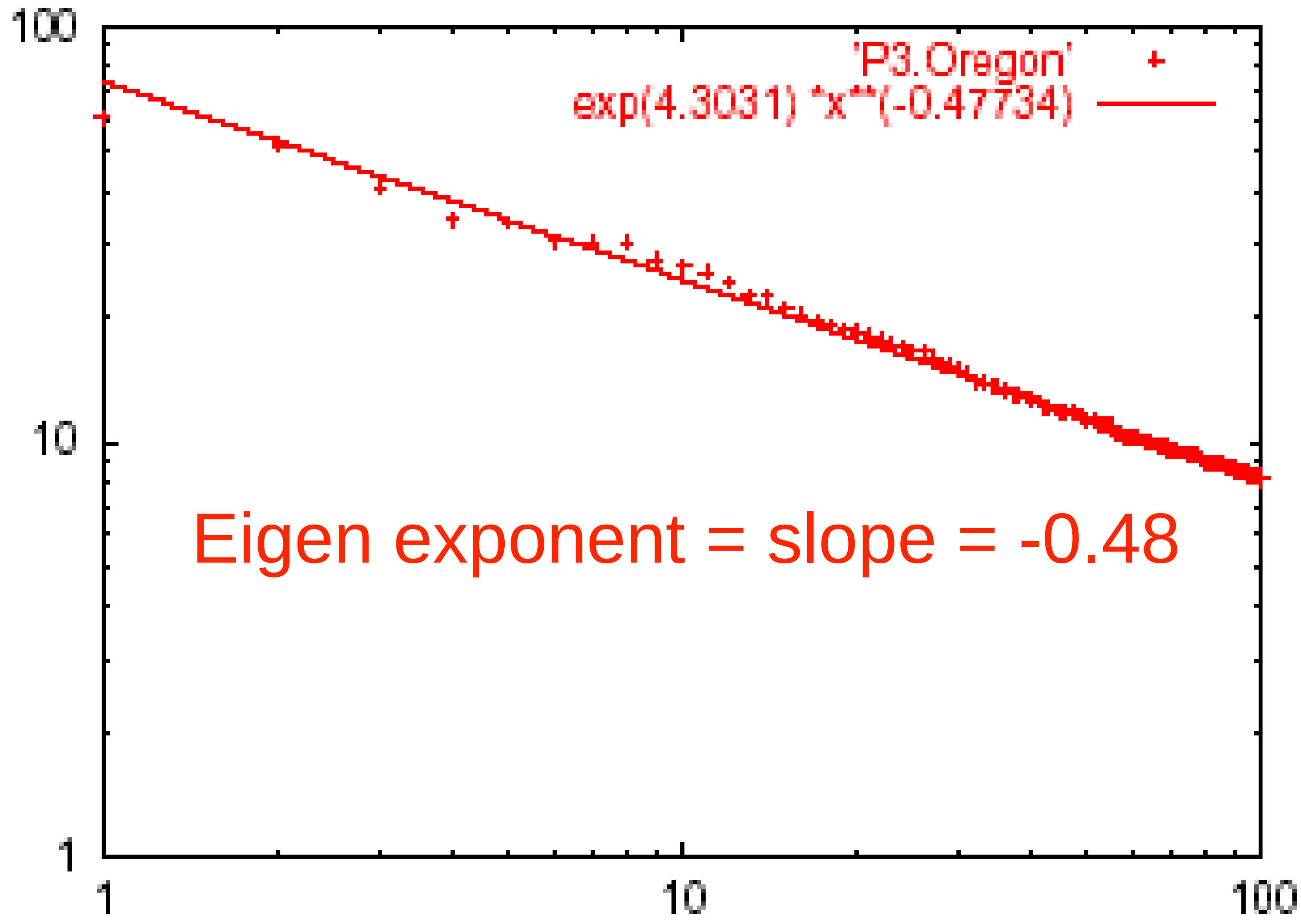
But: triangles are expensive to compute
(3-way join; several approx. algos)
Q: Can we do that quickly?
A: Yes!

$$\#triangles = 1/6 \; Sum \; (\lambda_i^3)$$

(and, because of skewness,
we only need the top few eigenvalues!

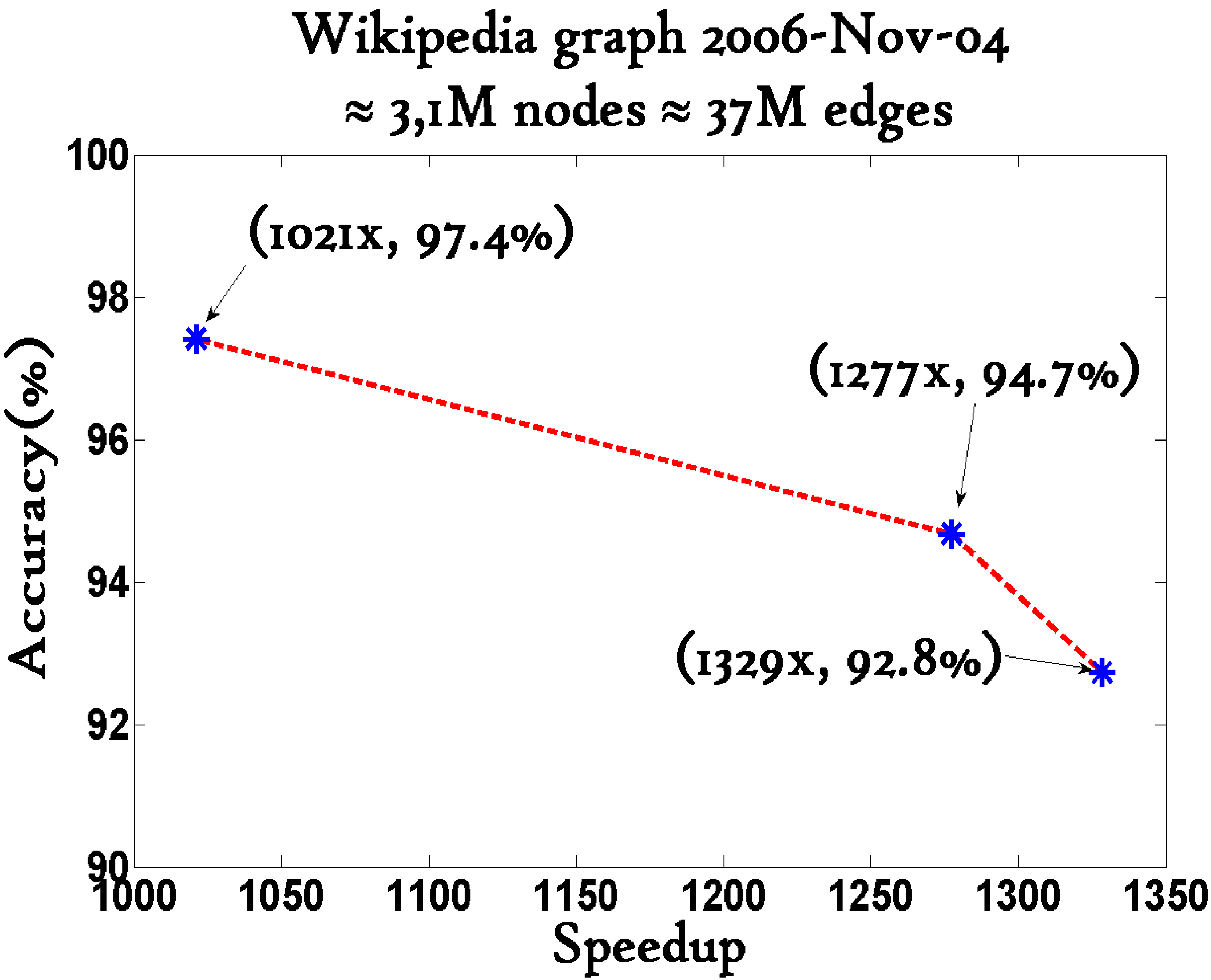
Power Law in Eigenvalues of Adjacency Matrix

Eigenvalue



Eigen exponent = slope = -0.48

Rank of decreasing eigenvalue



1000x+ speed-up, >90% accuracy

More Centrality Measures…

- Degree
- Betweenness
- Closeness, by computing
  - Shortest paths
  - "**Proximity**" (usually via *random walks*) — **used successfully in a lot of applications**
- Eigenvector
- …

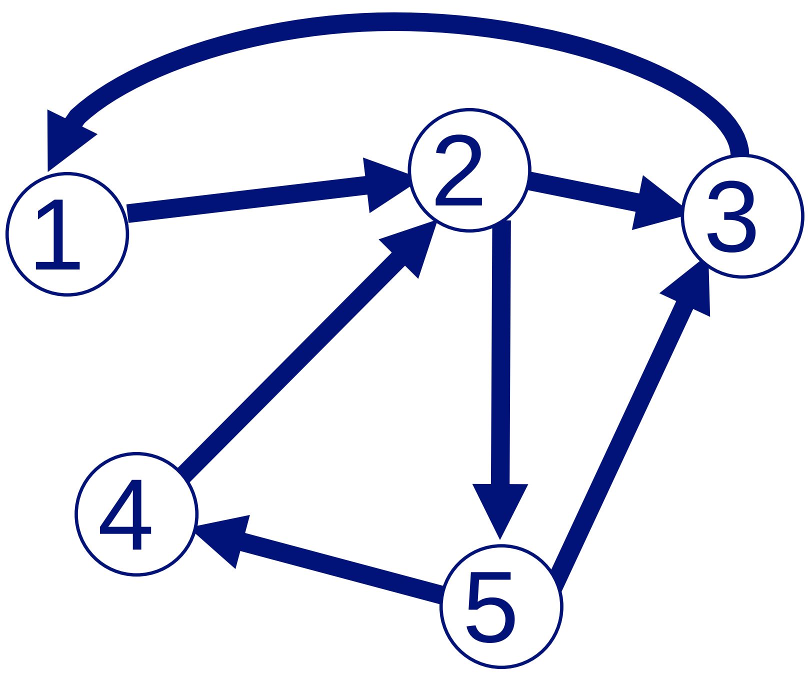
PageRank (Google)

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.

PageRank: Problem

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

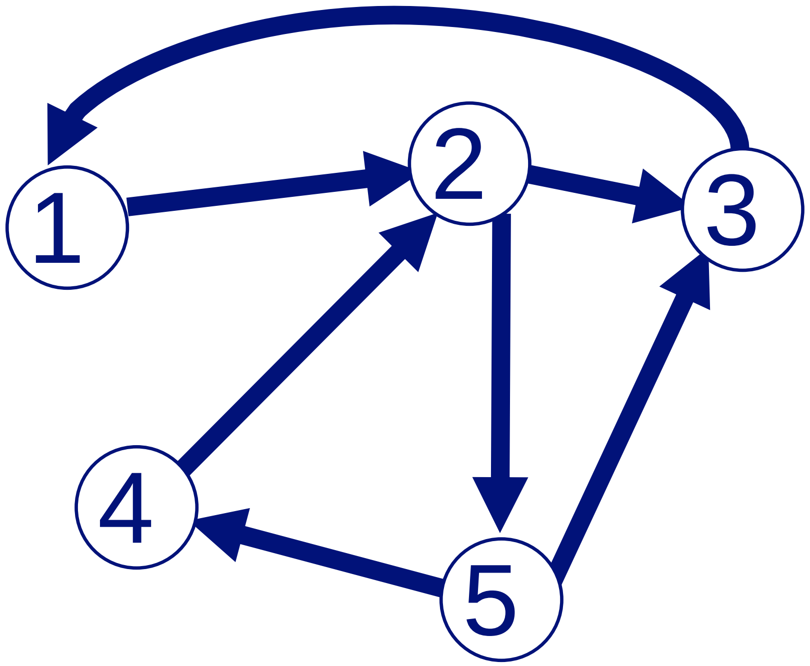


A node is important,
if it is connected
with important nodes
(recursive, but OK!)

PageRank: Solution

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

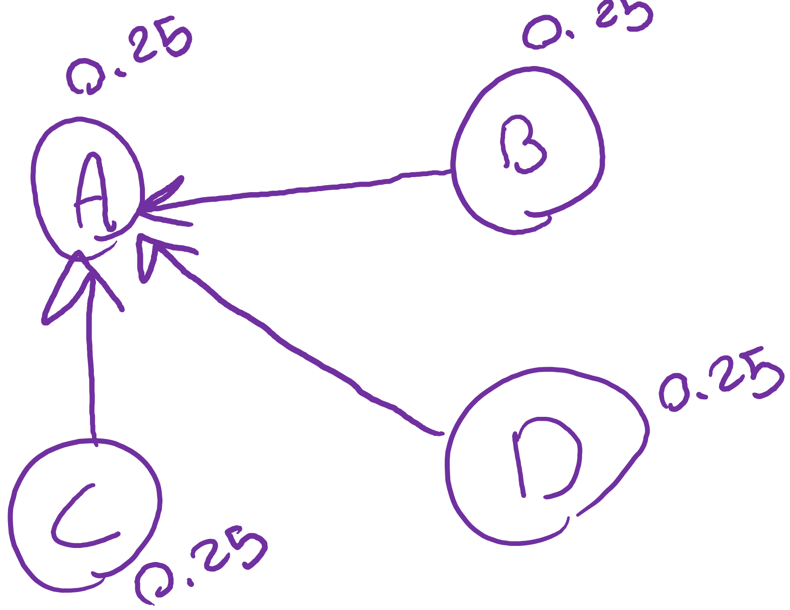
Proposed solution: use **random walk**; most "popular" nodes are the ones with highest steady state probability (ssp)



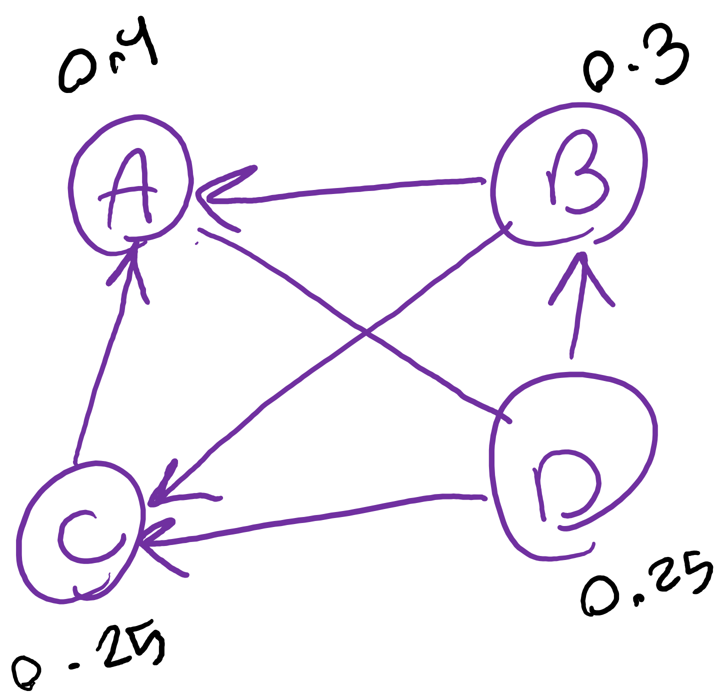
"state" = webpage

A node is important, if it is connected with important nodes (recursive, but OK!)

$\frac{1}{4} = \cancel{0.25}$



$$PR(A) = \cancel{PR(B)} + PR(C) + PR(D)$$
$$= 0.25 + 0.25 + 0.25 = 0.75$$

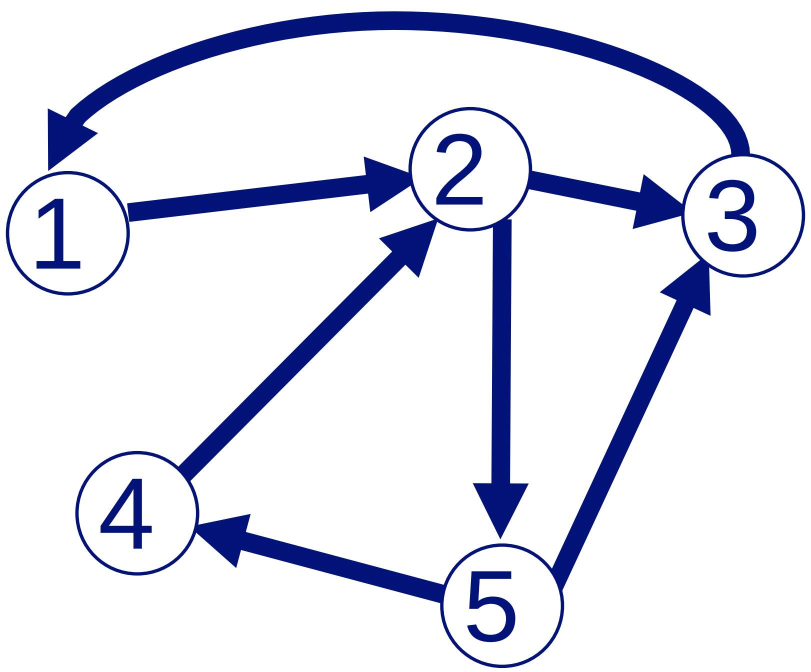


$$PR(A) = \frac{0.25}{2} + 0.25 + \frac{0.25}{3} = 0.4$$

$$\cancel{PR}(A) = \frac{PR(B)}{2} + PR(C) + \frac{PR(D)}{3}$$

(Simplified) PageRank

Let **B** be the transition matrix: transposed, column-normalized



From

To

$$\underbrace{\begin{bmatrix} & & 1 & & \\ 1 & & & 1 & \\ & 1/2 & & & 1/2 \\ & & & & 1/2 \\ & 1/2 & & & \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix}}_{\mathbf{p}} = \underbrace{\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix}}_{\mathbf{p}}$$

How to compute SSP:
https://fenix.tecnico.ulisboa.pt/downloadFile/3779579688473/6.3.pdf
http://www.sosmath.com/matrix/markov/markov.html

(Simplified) PageRank

$A\underset{\sim}{x} = \lambda \underset{\sim}{x}$

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

- In short: imagine a person **randomly moving** along the edges/links
- A node's PageRank score is the steady-state probability (ssp) of finding the person at that node

Full version of algorithm:

With **occasional random jumps to any nodes**

Why? To make the matrix **irreducible.**

Irreducible = from any state (node), there's non-zero probability to reach any other state (node)

Full Algorithm

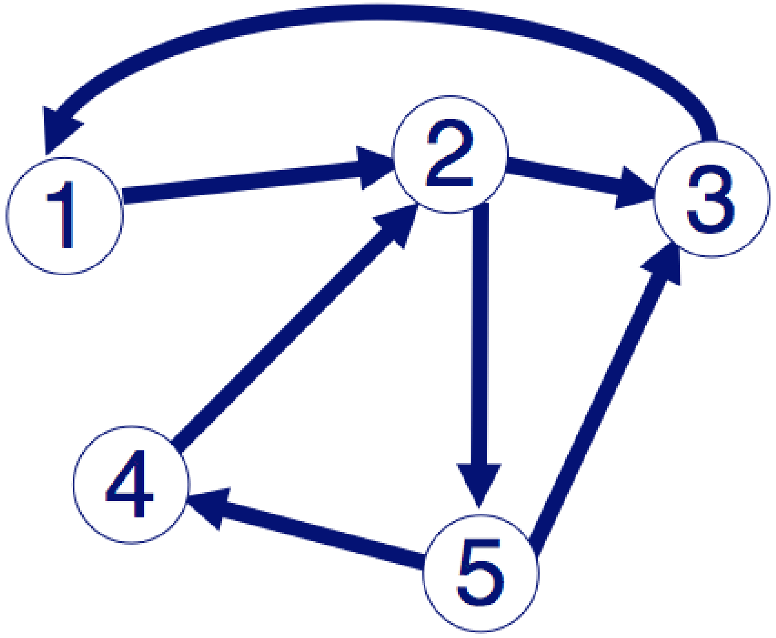
$p = Bp$

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

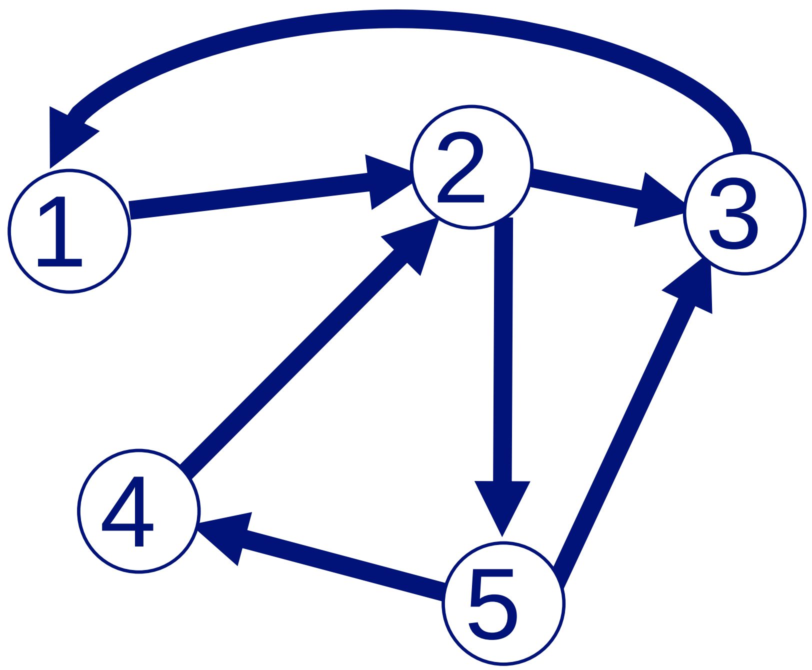
Then, we have

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

$$\begin{bmatrix} 1/n \\ 1/n \\ 1/n \\ 1/n \\ 1/n \end{bmatrix}$$



How to compute PageRank for huge matrix?



Use the power iteration method

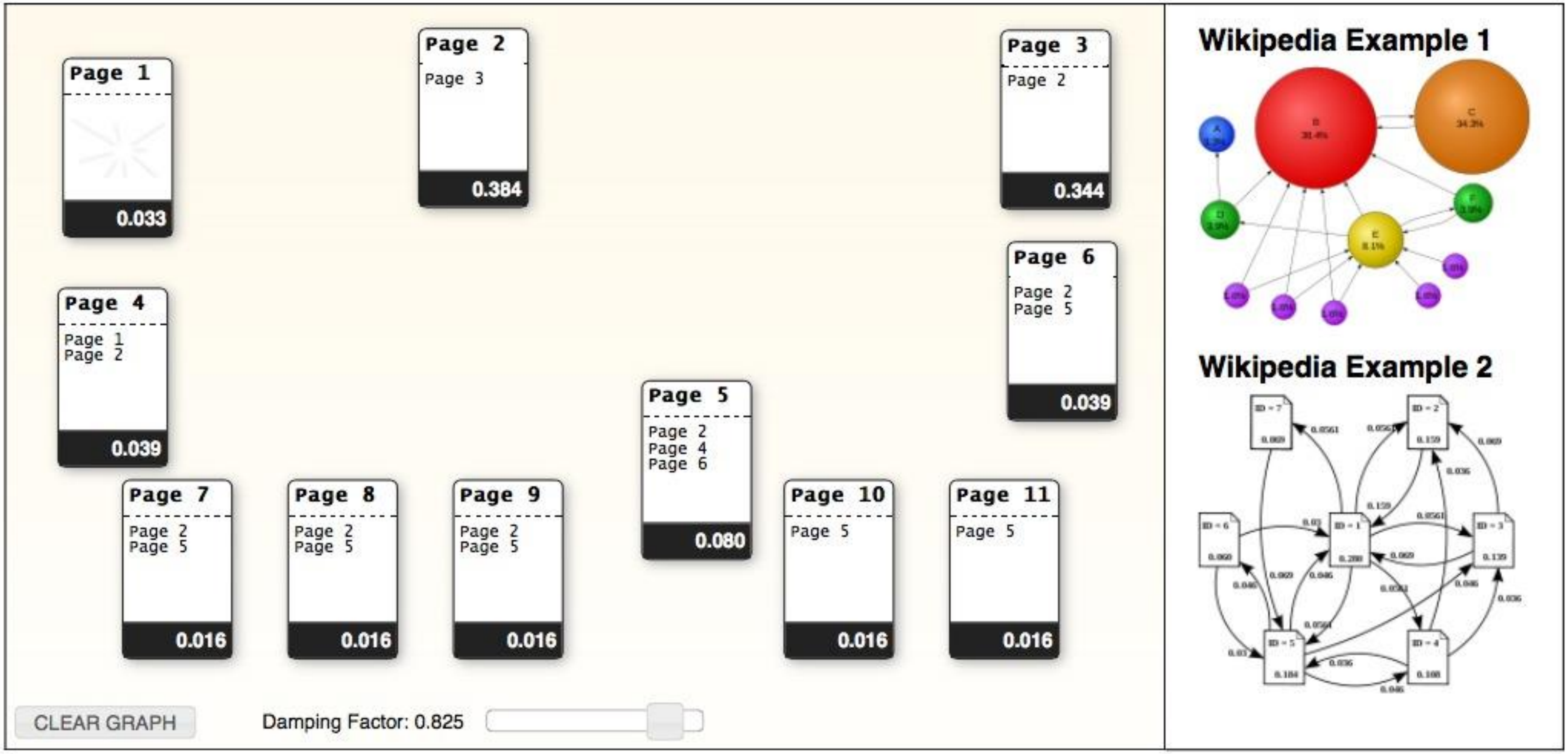
http://en.wikipedia.org/wiki/Power_iteration

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

$$\underbrace{\begin{bmatrix} p1 \\ p2 \\ p3 \\ p4 \\ p5 \end{bmatrix}}_{\mathbf{p'}} = c \underbrace{\begin{bmatrix} & & 1 & & \\ 1 & & & 1 & \\ & 1/2 & & & 1/2 \\ & & & & 1/2 \\ & 1/2 & & & \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} p1 \\ p2 \\ p3 \\ p4 \\ p5 \end{bmatrix}}_{\mathbf{p}} + \frac{(1-c)}{n} \begin{bmatrix} 1 \\ 1 \\ \cdots \\ 1 \end{bmatrix}$$

Can initialize this vector to any non-zero vector, e.g., all “1”s

PageRank Explained with Javascript



Also great for checking the correctness of your PageRank Implementation.

http://www.cs.duke.edu/csed/principles/pagerank/

PageRank for graphs (generally)

You can run PageRank on **any graphs**

- All you need are the **graph edges**!

Should be in your algorithm “toolbox”

- Better than degree centrality
- Fast to compute for large graphs, runtime linear in the number of edges, O(E)

But can be “misled” (Google Bomb)

- How?

Personalized PageRank

Intuition: not all pages are equal, some more relevant to some people

Goal: rank pages in a way that those more relevant to you will be ranked higher

How? Make just **one** small change to PageRank

Personalized PageRank

With probability 1-c, fly-out to

~~a random node~~ **some preferred nodes**

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

$$\begin{bmatrix} p'_1 \\ p'_2 \\ p'_3 \\ p'_4 \\ p'_5 \end{bmatrix} = 0.8 \begin{bmatrix} & & 1 & & \\ 1 & & & 1 & \\ & 1/2 & & & 1/2 \\ & & & & 1/2 \\ & 1/2 & & & \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix} + \frac{0.2}{5} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ \mathbf{1} \\ 0 \\ 0 \\ \mathbf{1} \end{bmatrix}$$

Default value for c

Can initialize this vector to any non-zero vector, e.g., all “1”s

Why Learn Personalized PageRank?

For **recommendation**

- If I like webpage A, what else do I like?
- If I bought product A, what other products would I also buy?

Visualizing and interacting with large graphs

- Instead of visualizing every single nodes, visualize the **most important ones**

Very flexible — works on any graph

Related "guilt-by-association" / diffusion techniques

- **Personalized PageRank**
  (= Random Walk with Restart)
- "Spreading activation" or "degree of interest" in Human-Computer Interaction (HCI)
- Belief Propagation
  (powerful inference algorithm, for fraud detection, image segmentation, error-correcting codes, etc.)

Why are these algorithms popular?

- **Intuitive to interpret**
  uses “network effect”, homophily
- **Easy to implement**
  math is relatively simple (mainly matrix-vector multiplication)
- **Fast**
  run time linear to #edges, or better
- **Probabilistic** meaning

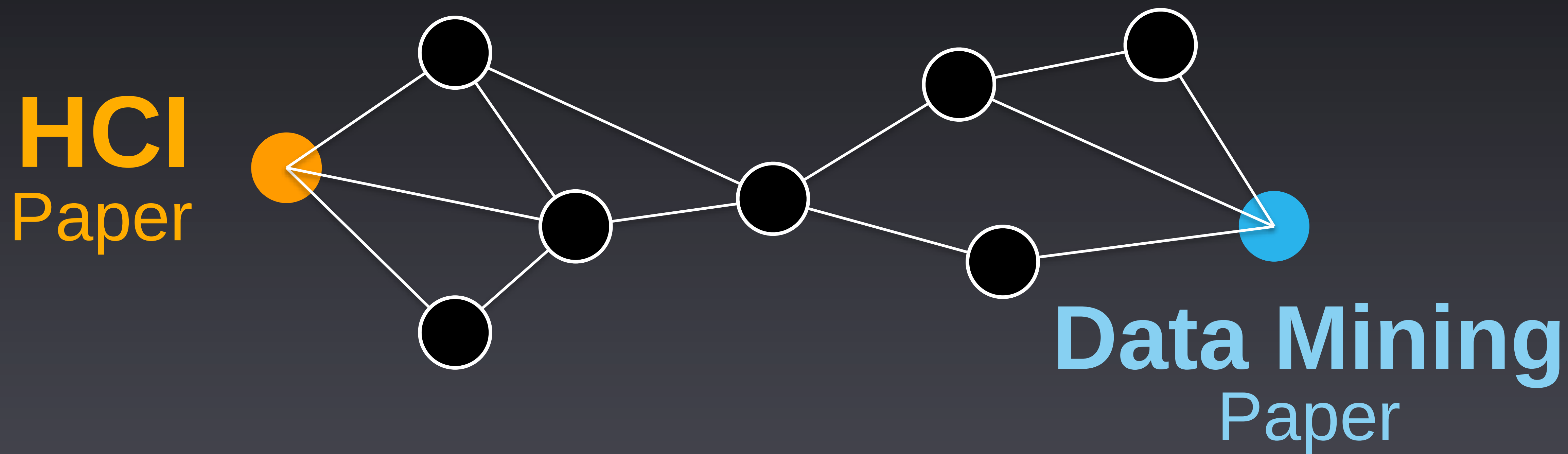
Human-In-The-Loop Graph Mining

Apolo:
Machine Learning + Visualization

CHI 2011

Apolo: Making Sense of Large Network Data by Combining Rich User Interaction and Machine Learning

Finding More Relevant Nodes



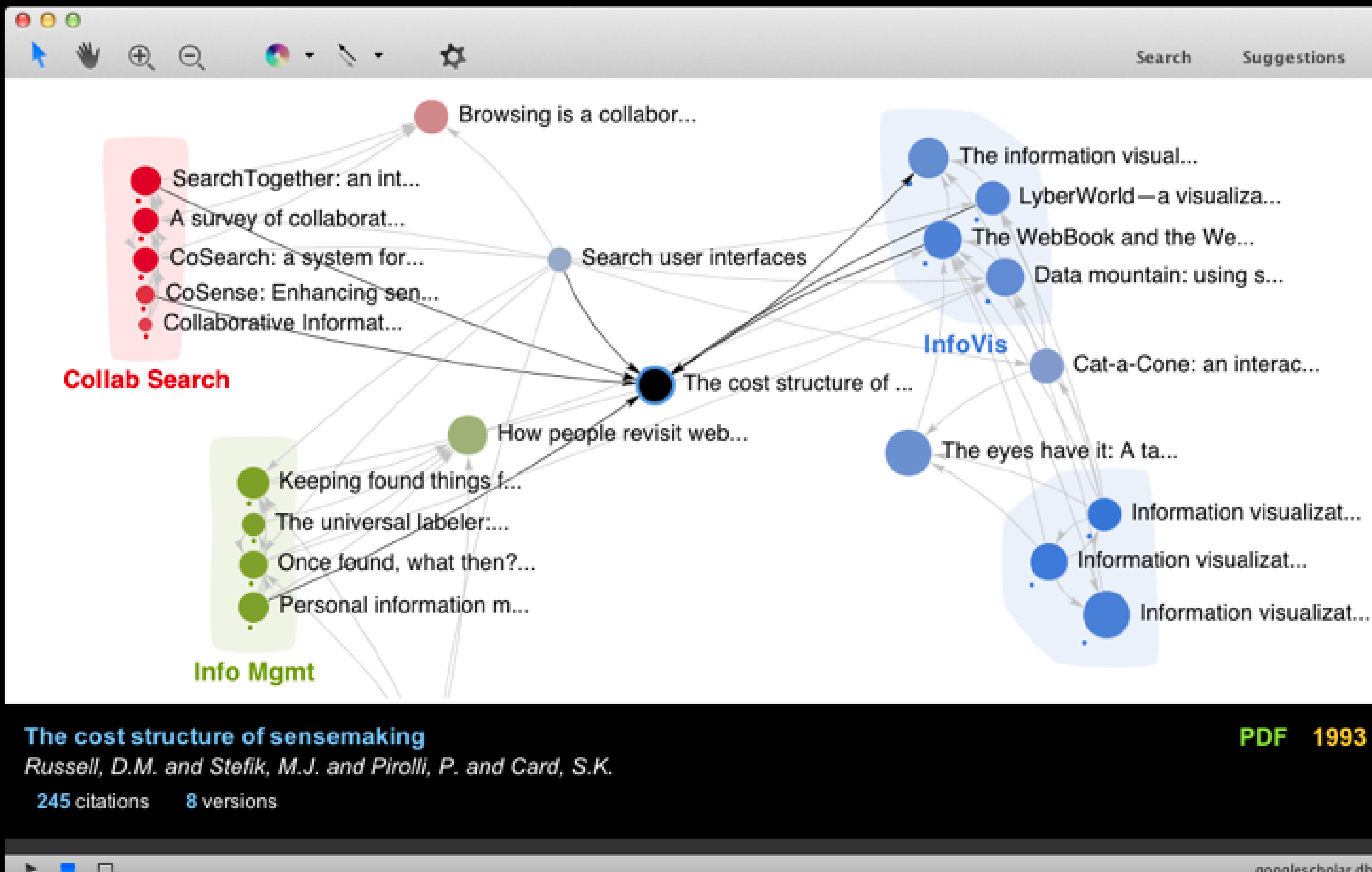
Citation network

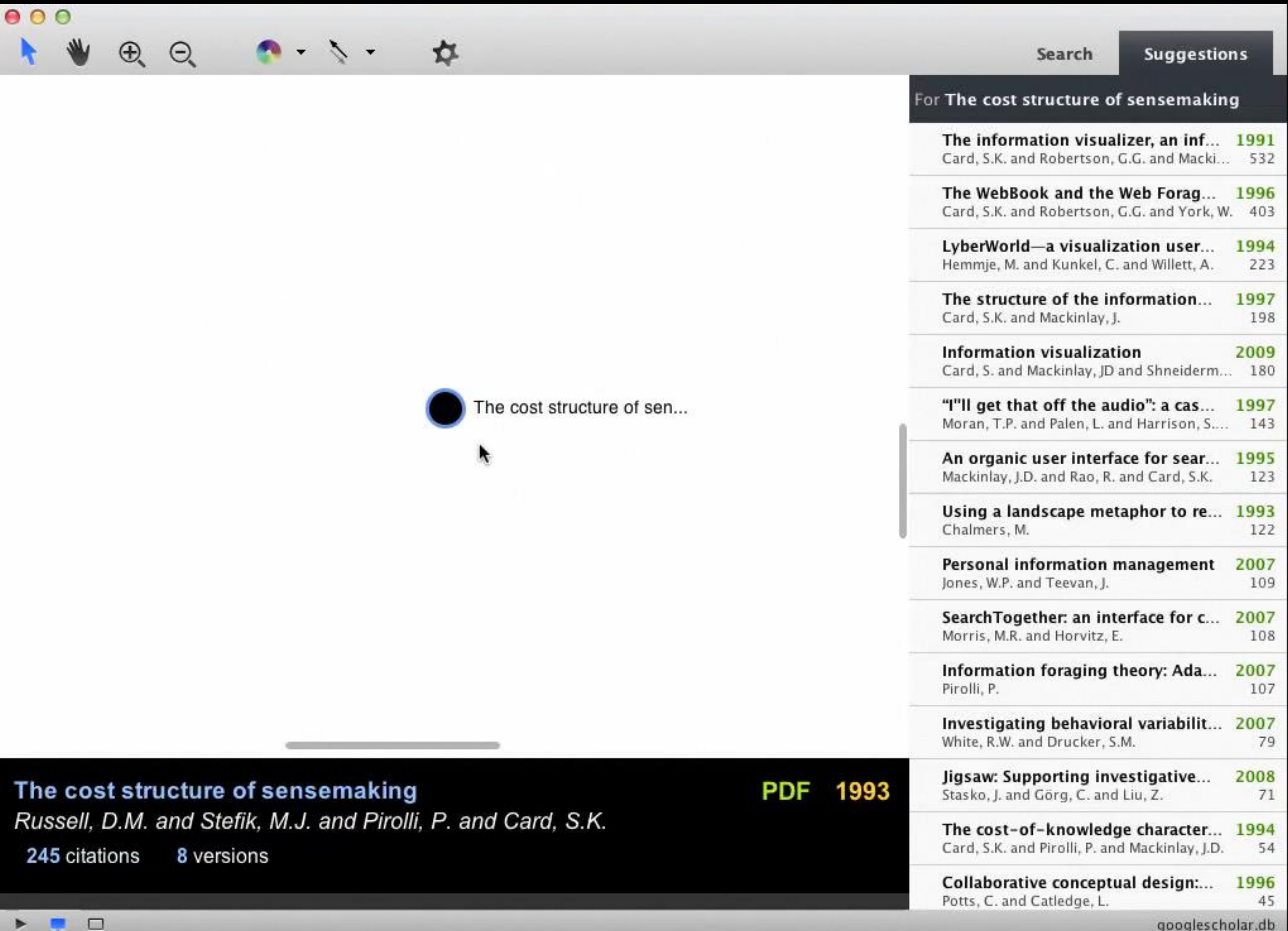
Apolo uses **guilt-by-association**

(Belief Propagation, similar to personalized PageRank)

Demo: Mapping the Sensemaking Literature

Nodes: 80k papers from Google Scholar (node size: #citation)
Edges: 150k citations

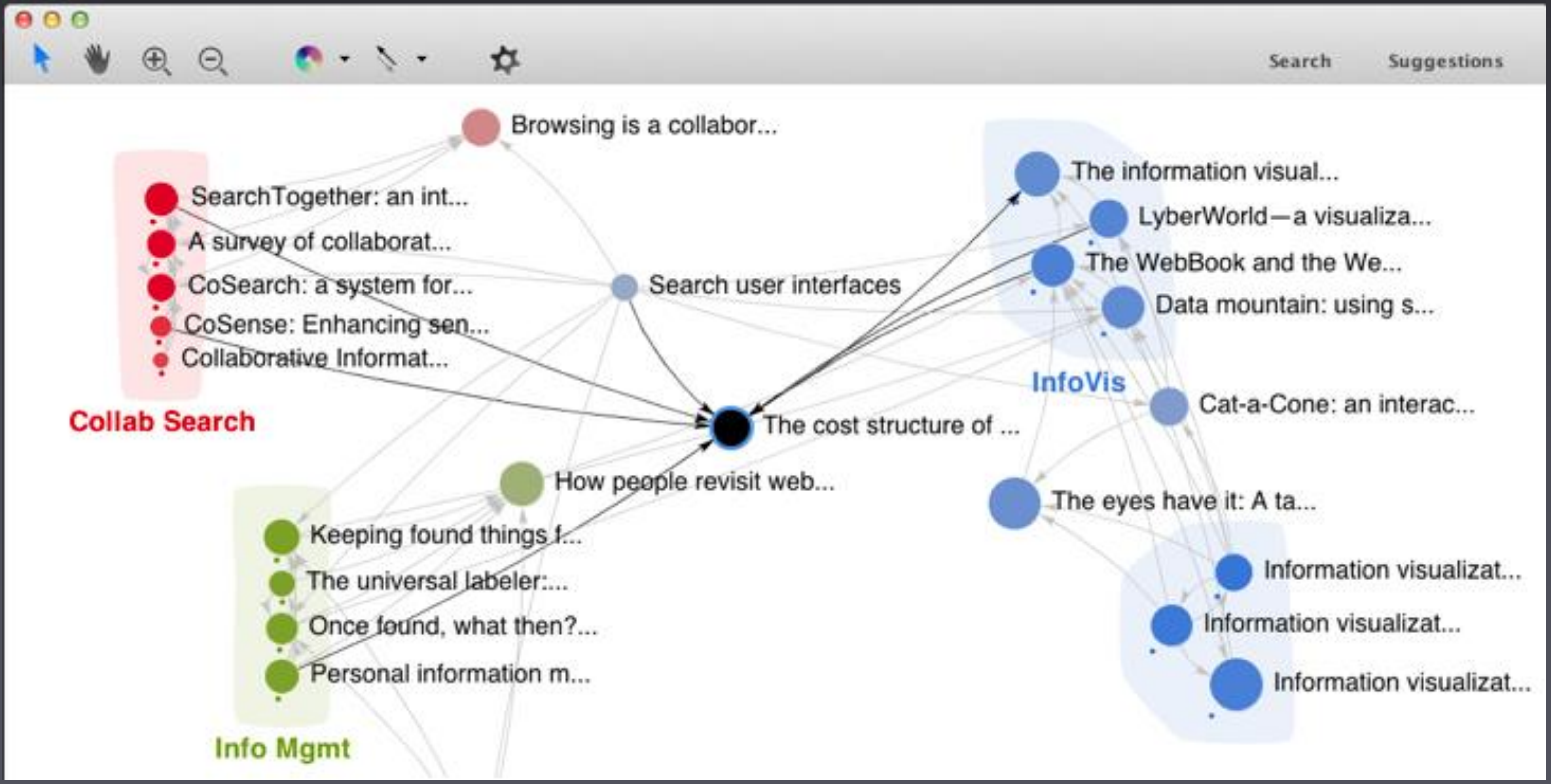



Search
Suggestions
For The cost structure of sensemaking
The information visualizer, an inf... 1991
Card, S.K. and Robertson, G.G. and Macki... 532
The WebBook and the Web Forag... 1996
Card, S.K. and Robertson, G.G. and York, W. 403
LyberWorld—a visualization user... 1994
Hemmje, M. and Kunkel, C. and Willett, A. 223
The structure of the information... 1997
Card, S.K. and Mackinlay, J. 198
Information visualization 2009
Card, S. and Mackinlay, JD and Shneiderm... 180
"I"ll get that off the audio": a cas... 1997
Moran, T.P. and Palen, L. and Harrison, S.... 143
An organic user interface for sear... 1995
Mackinlay, J.D. and Rao, R. and Card, S.K. 123
Using a landscape metaphor to re... 1993
Chalmers, M. 122
Personal information management 2007
Jones, W.P. and Teevan, J. 109
SearchTogether: an interface for c... 2007
Morris, M.R. and Horvitz, E. 108
Information foraging theory: Ada... 2007
Pirolli, P. 107
Investigating behavioral variabilit... 2007
White, R.W. and Drucker, S.M. 79
Jigsaw: Supporting investigative... 2008
Stasko, J. and Görg, C. and Liu, Z. 71
The cost-of-knowledge character... 1994
Card, S.K. and Pirolli, P. and Mackinlay, J.D. 54
Collaborative conceptual design:... 1996
Potts, C. and Catledge, L. 45
The cost structure of sen...
The cost structure of sensemaking
PDF 1993
Russell, D.M. and Stefik, M.J. and Pirolli, P. and Card, S.K.
245 citations 8 versions
googlescholar.db

Key Ideas (Recap)

Specify exemplars

Find other relevant nodes (BP)



Apolo's Contributions

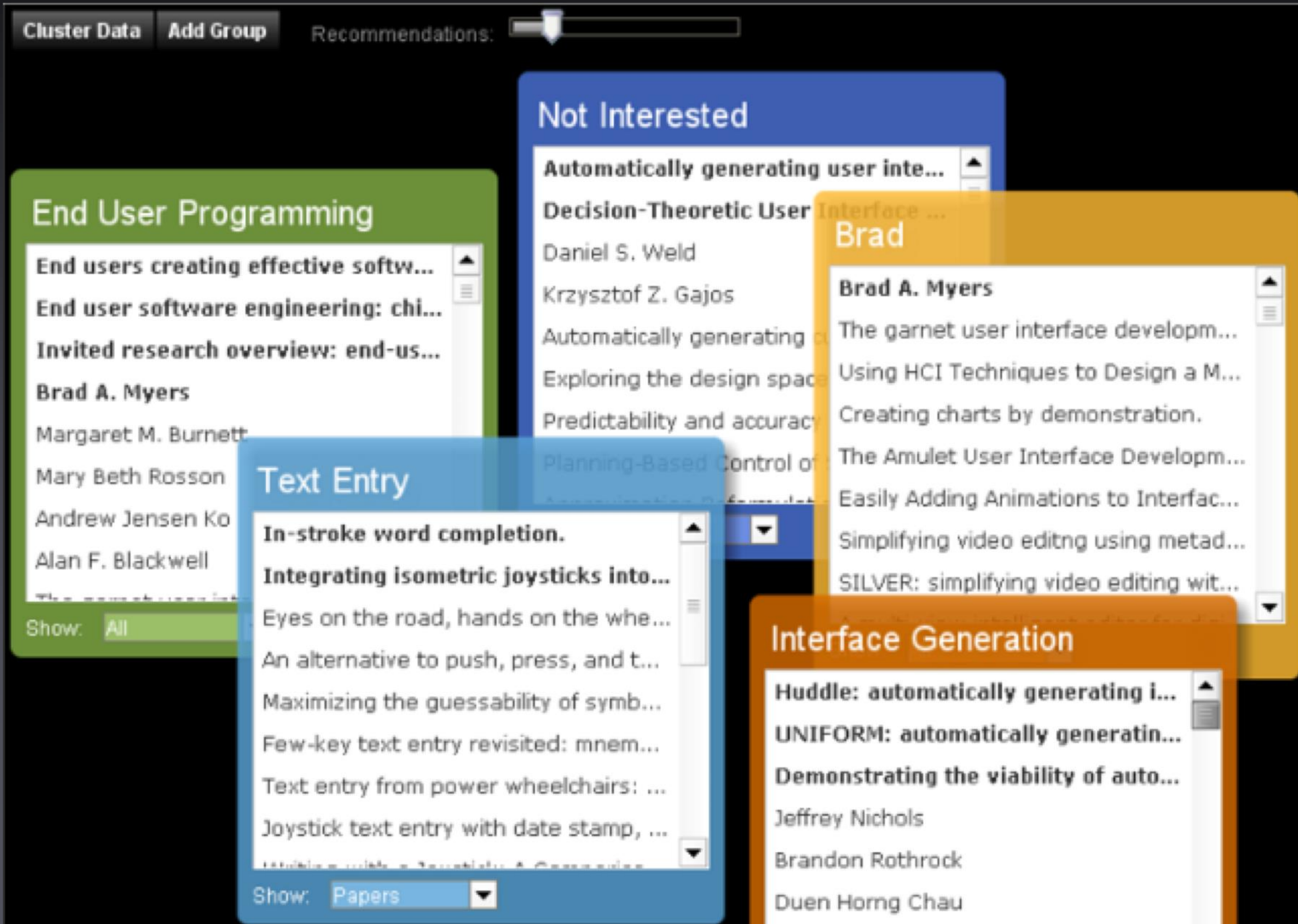
1 **Human** + **Machine**

It was like having a **partnership** with the machine.

Apolo User

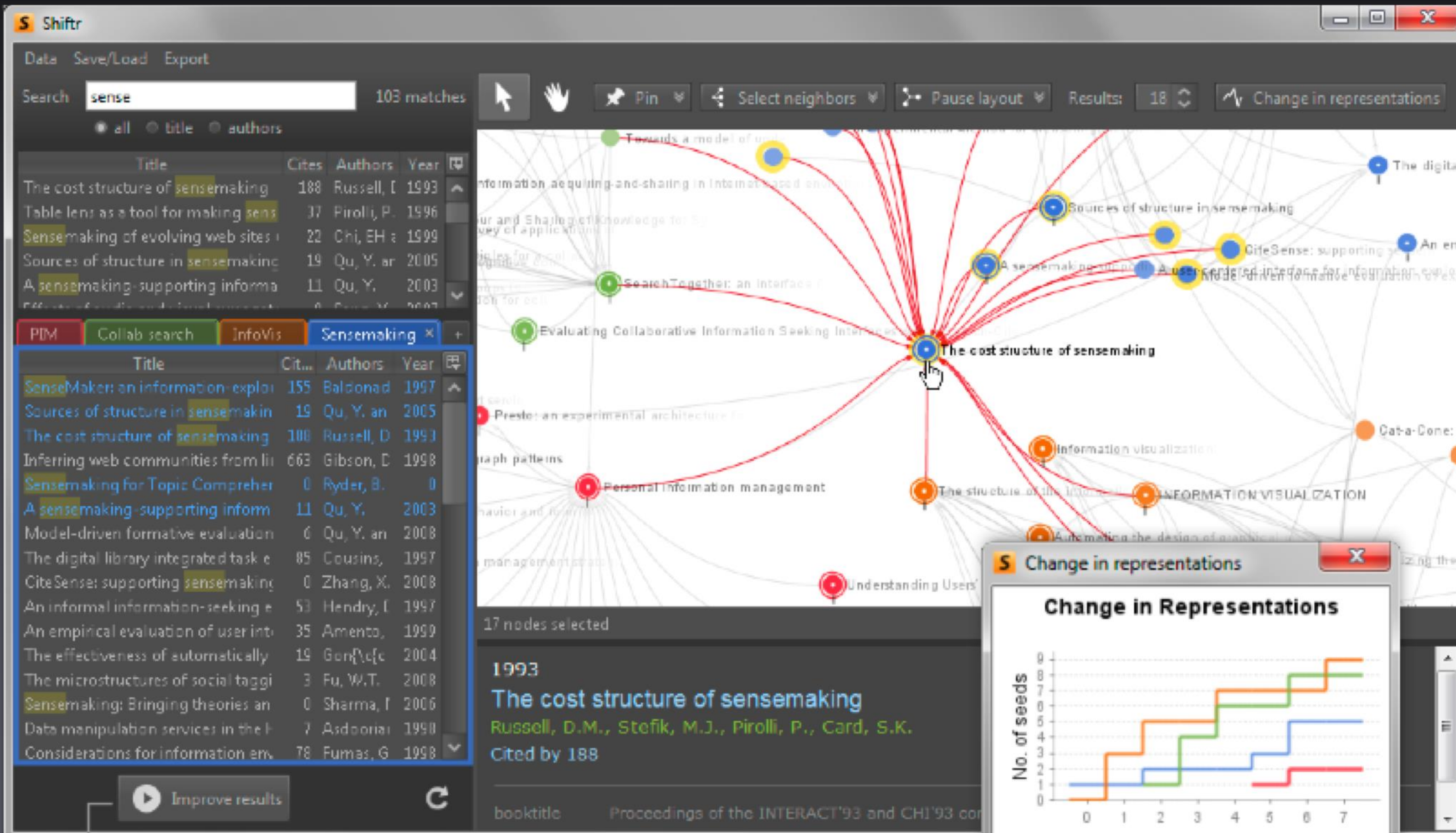
2 **Personalized Landscape**

Apolo 2009



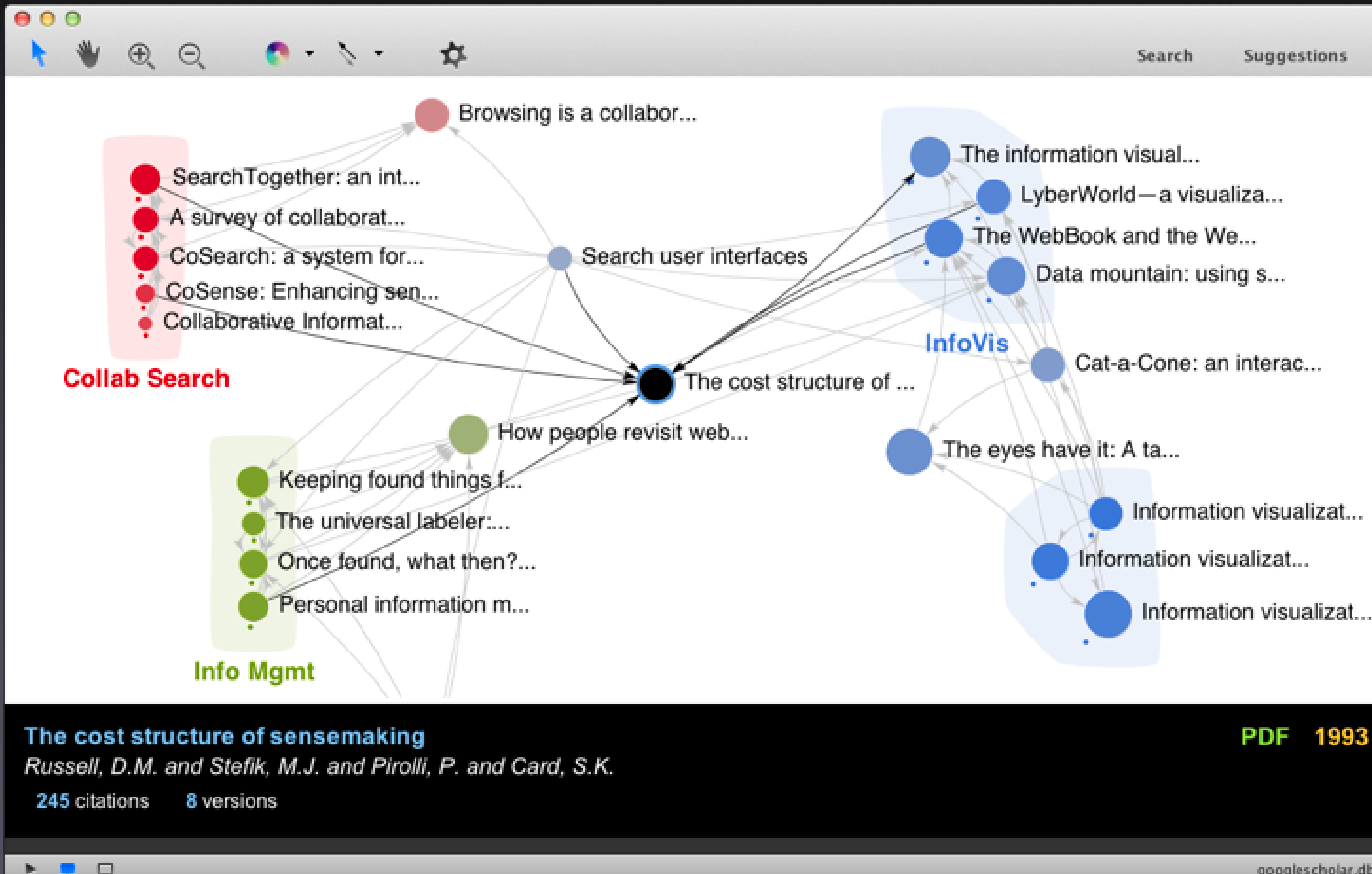
Cluster Data
Add Group
Recommendations:
Not Interested
Automatically generating user inte...
Decision-Theoretic User
Daniel S. Weld
Krzysztof Z. Gajos
Automatically generating
Exploring the design spac
Predictability and accurac
End User Programming
End users creating effective softw...
End user software engineering: chi...
Invited research overview: end-us...
Brad A. Myers
Margaret M. Burnett
Mary Beth Rosson
Andrew Jensen Ko
Alan F. Blackwell
Show:
All
Brad
Brad A. Myers
The garnet user interface developm...
Using HCI Techniques to Design a M...
Creating charts by demonstration.
The Amulet User Interface Developm...
Easily Adding Animations to Interfac...
Simplifying video editng using metad...
SILVER: simplifying video editing wit...
Text Entry
In-stroke word completion.
Integrating isometric joysticks into...
Eyes on the road, hands on the whe...
An alternative to push, press, and t...
Maximizing the guessability of symb...
Few-key text entry revisited: mnem...
Text entry from power wheelchairs: ...
Joystick text entry with date stamp, ...
Show:
Papers
Interface Generation
Huddle: automatically generating i...
UNIFORM: automatically generatin...
Demonstrating the viability of auto...
Jeffrey Nichols
Brandon Rothrock
Duen Horng Chau

Apolo 2010


Shiftr
Data Save/Load Export
Search sense 103 matches
all title authors
Pin Select neighbors Pause layout Results: 18 Change in representations
PIM Collab search InfoVis Sensemaking
The cost structure of sensemaking
17 nodes selected
1993
The cost structure of sensemaking
Russell, D.M., Stefik, M.J., Pirolli, P., Card, S.K.
Cited by 188
booktitle
Improve results
Change in representations
Change in Representations
No. of seeds

Apolo 2011

22,000 lines of code. Java 1.6. Swing.
Uses SQLite3 to store graph on disk

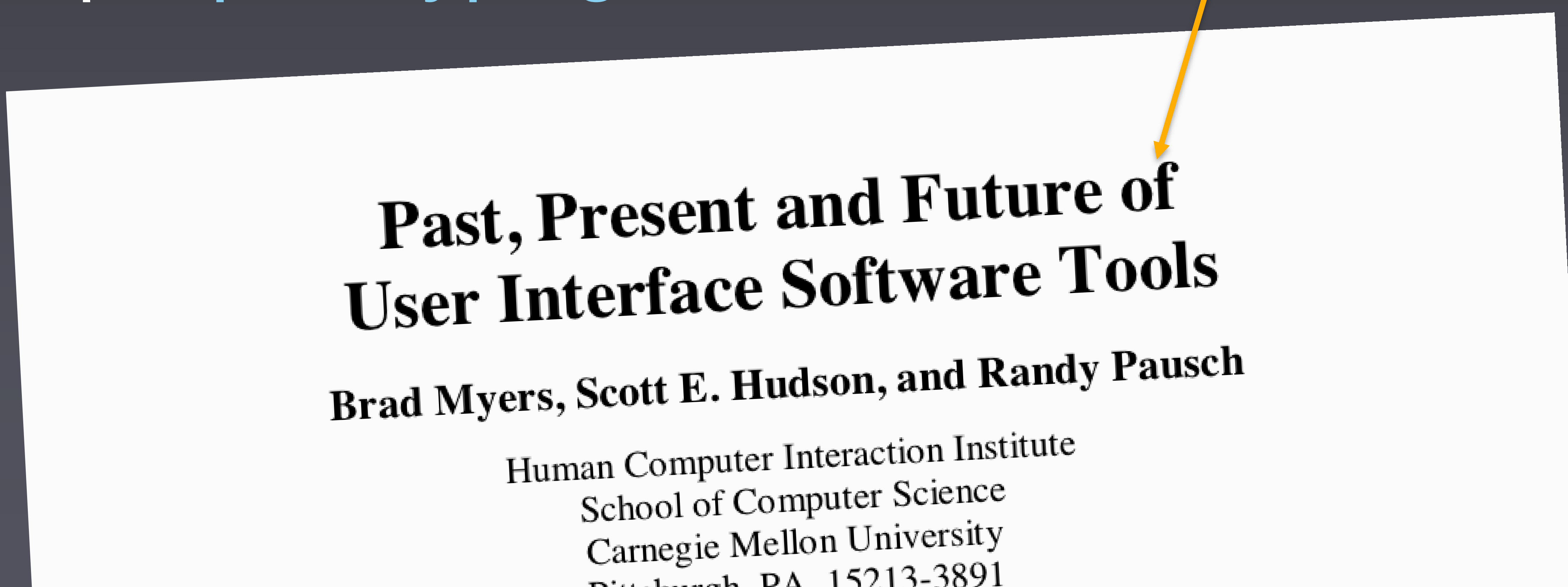


User Study

Used citation network

<u>Task</u>: Find related papers for 2 sections in a survey paper on *user interface*

- Model-based generation of UI
- Rapid prototyping tools



Past, Present and Future of User Interface Software Tools

Brad Myers, Scott E. Hudson, and Randy Pausch

Human Computer Interaction Institute
School of Computer Science
Carnegie Mellon University

Between subjects design

Participants: **grad student** or **research staff**

Judges' Scores

Higher is better.
Apolo wins.

** Statistically significant, by two-tailed t test, p <0.05*

Practitioners’ guide to building (interactive) applications

What kinds of **prototypes**?

- Paper prototype, lo-fi prototype, high-fi prototype

Important to involve **REAL users** as early as possible

- Recruit your friends to try your tools
- Lab study (controlled, as in Apolo)
- Longitudinal study (usage over months)
- Deploy it and see the world’s reaction!

- To learn more:
    - CS 6750 Human-Computer Interaction
    - CS 6455 User Interface Design and Evaluation

Practitioners' guide to building (interactive) applications

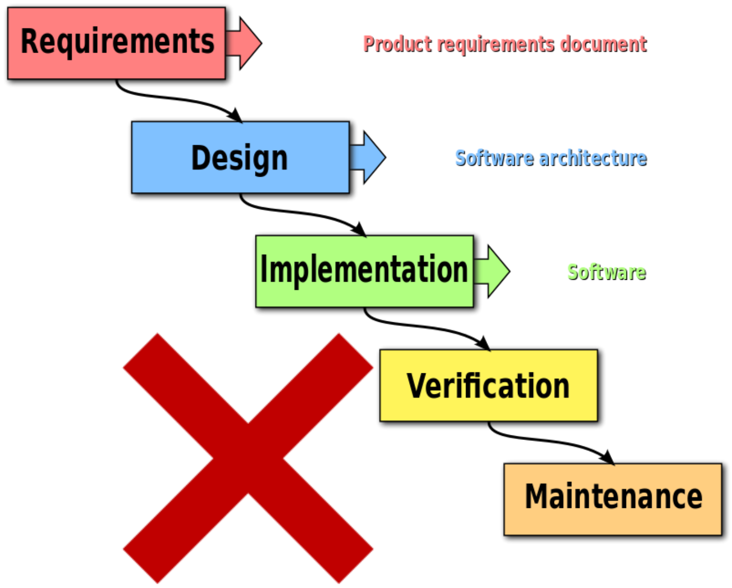
Think about scalability early

- Identify candidate scalable algorithms early on

Use **iterative** design approach, as in Apolo and industry

- Why? It's hard to get it right the first time
- Create prototype, evaluate, modify prototype, evaluate, ...
- Quick evaluation helps you identify **important fixes early — save you a lot of time overall**

Waterfall model
(software engineering)



If you want to know more about people…

http://amzn.com/0321767535