Chapter 8:
Graph Data

Part 1:
Link Analysis & Page Rank

Based on
Leskovec, Rajaraman, Ullman 2014:
Mining of Massive Datasets

- Exam on the 5th of February, 2016, 14.00 to 16.00

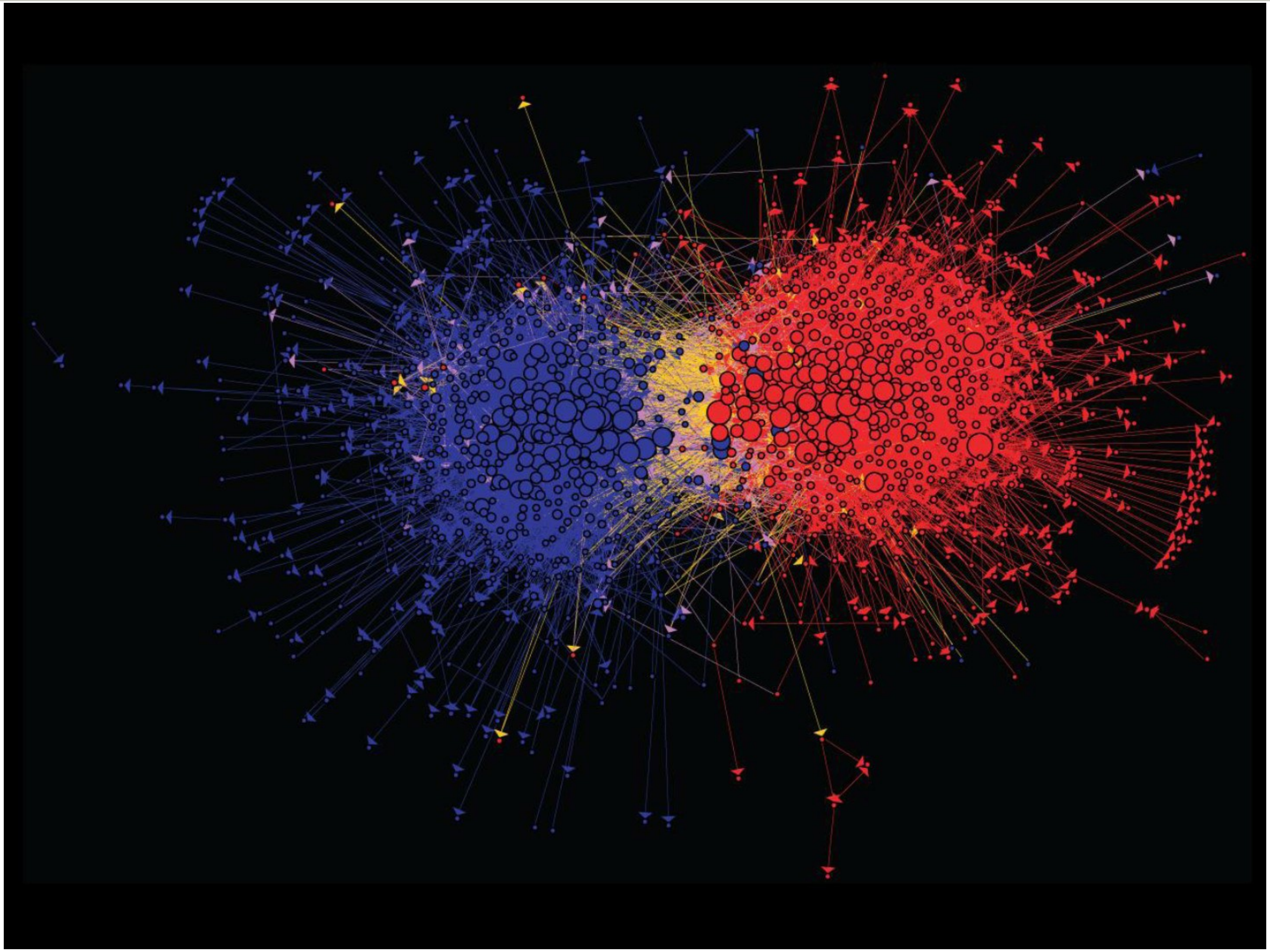
- If you wish to attend, please register!

http://uniworx.ifi.lmu.de/

Graph Data: Social Networks

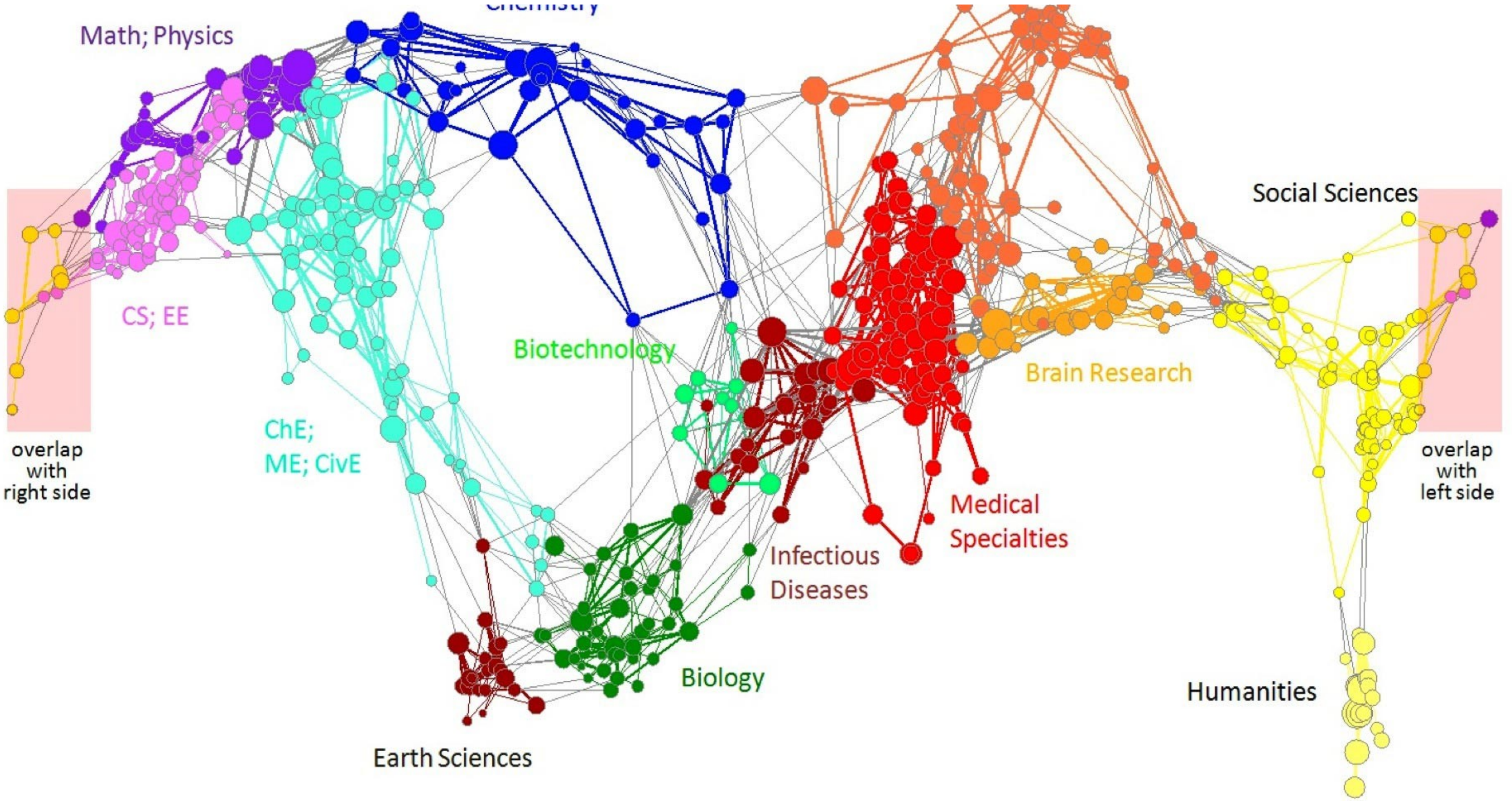
[Source: 4-degrees of separation, Backstrom-Boldi-Rosa-Ugander-Vigna, 2011]

Graph Data: Media Networks



Connections between political blogs
Polarization of the network [Adamic-Glance, 2005]

Graph Data: Information Networks

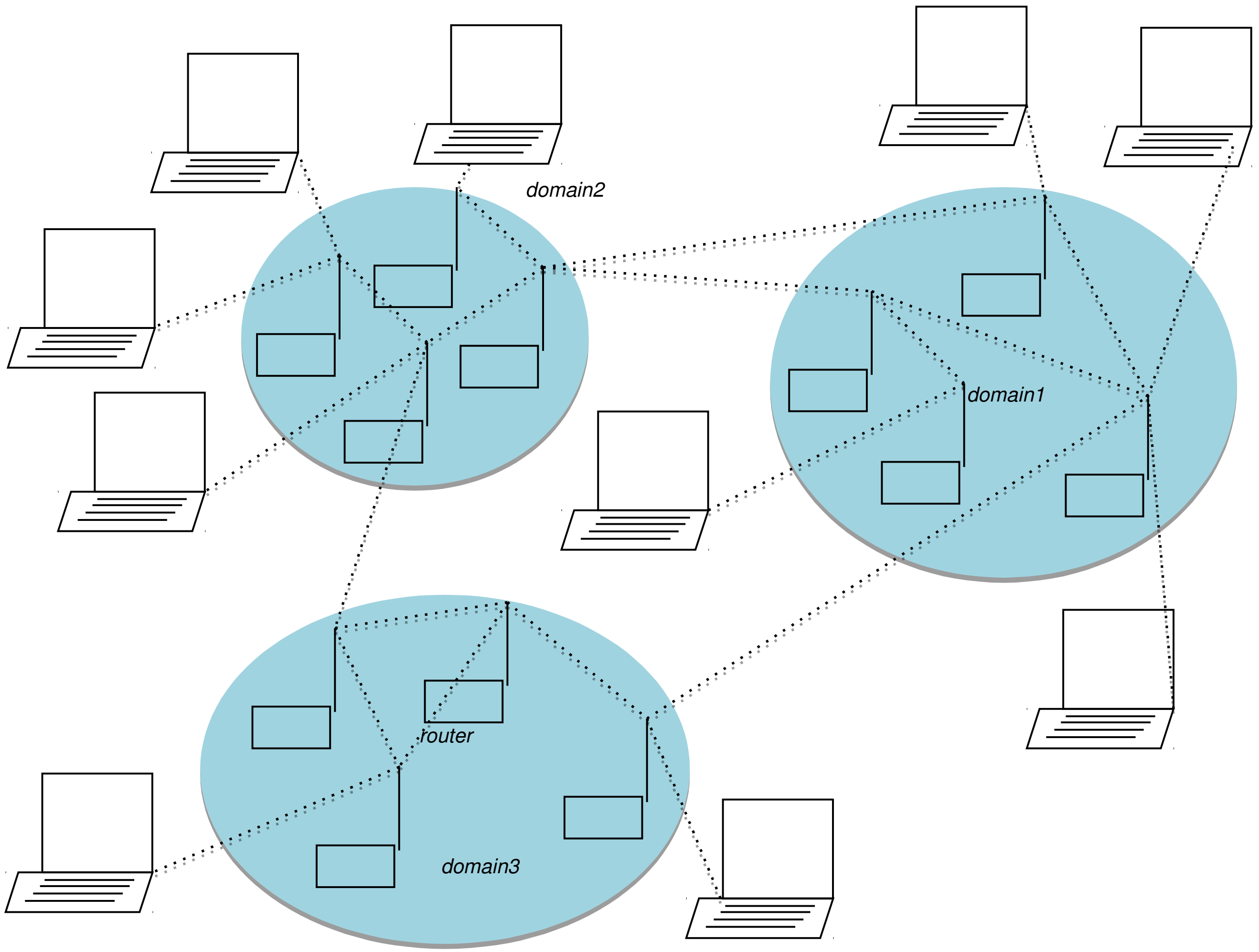


Citation Networks and Map of Science
[Börner et al., 2012]

Graph Data: Technological Networks

Road Network of Toulouse
[Mathieu Leplatre]

Graph Data: Communication Networks



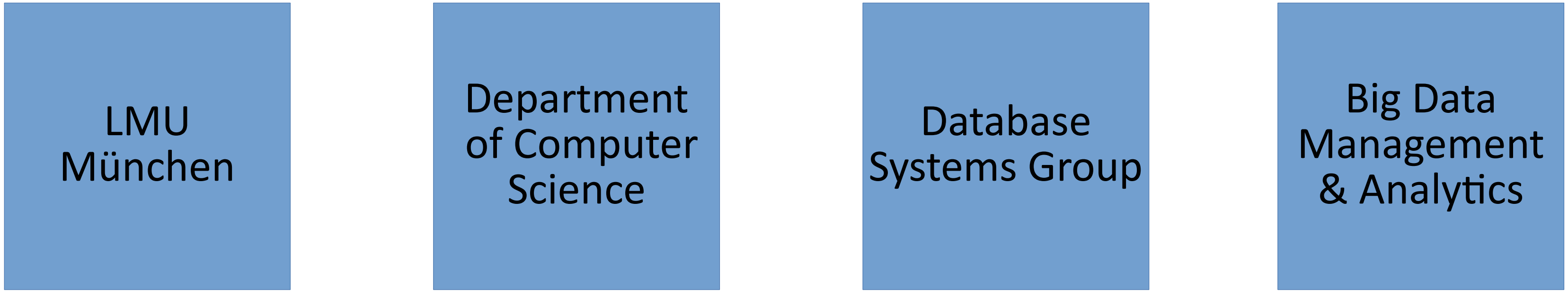
The Internet

DATABASE SYSTEMS GROUP

Web as a Graph

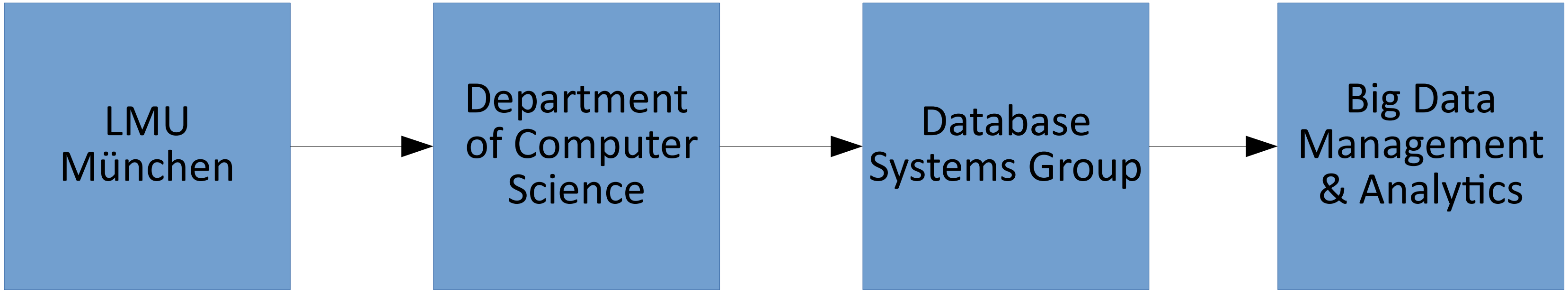
LMU

Web as a directed graph:
- Nodes: Webpages
- Edges: Hyperlinks

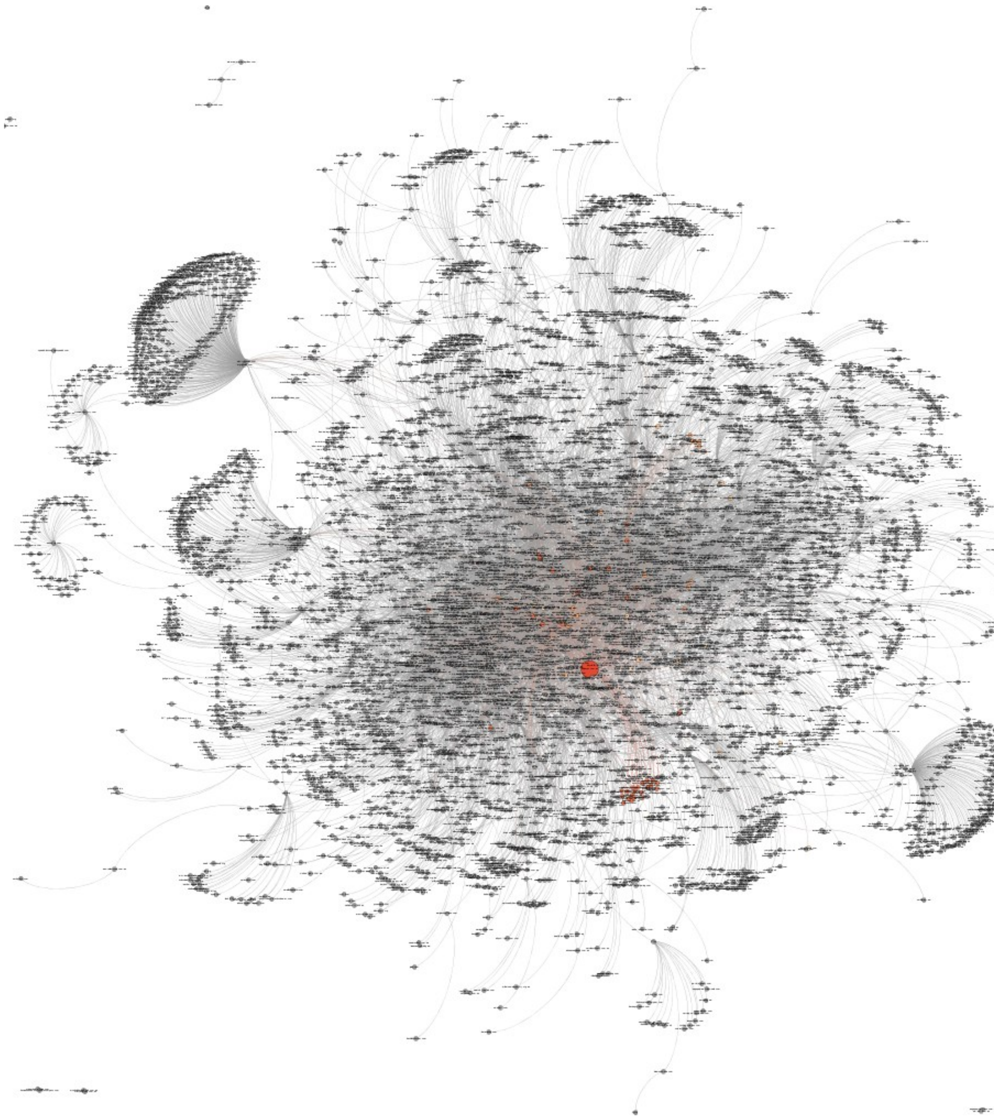


Web as a Graph

Web as a directed graph:
- Nodes: Webpages
- Edges: Hyperlinks



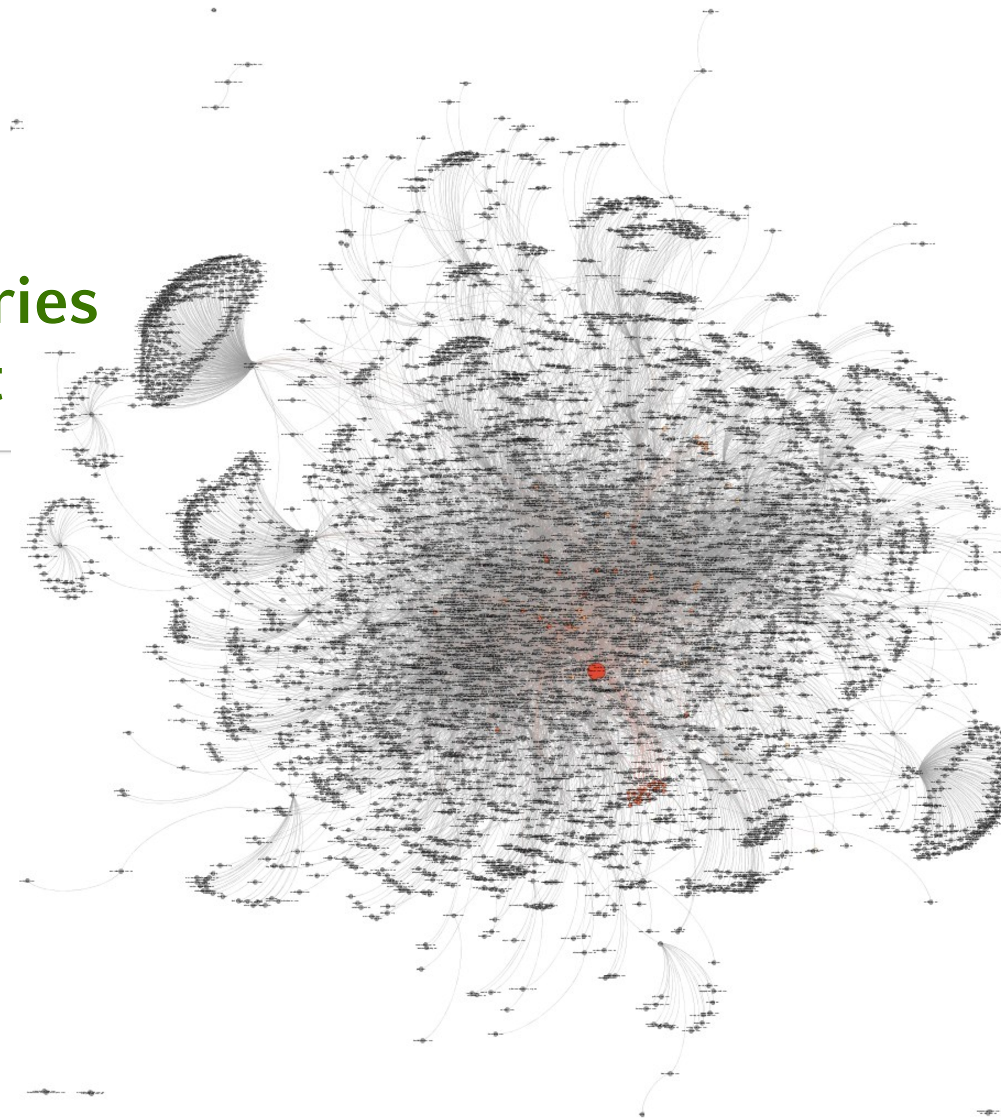
How to organise the web?



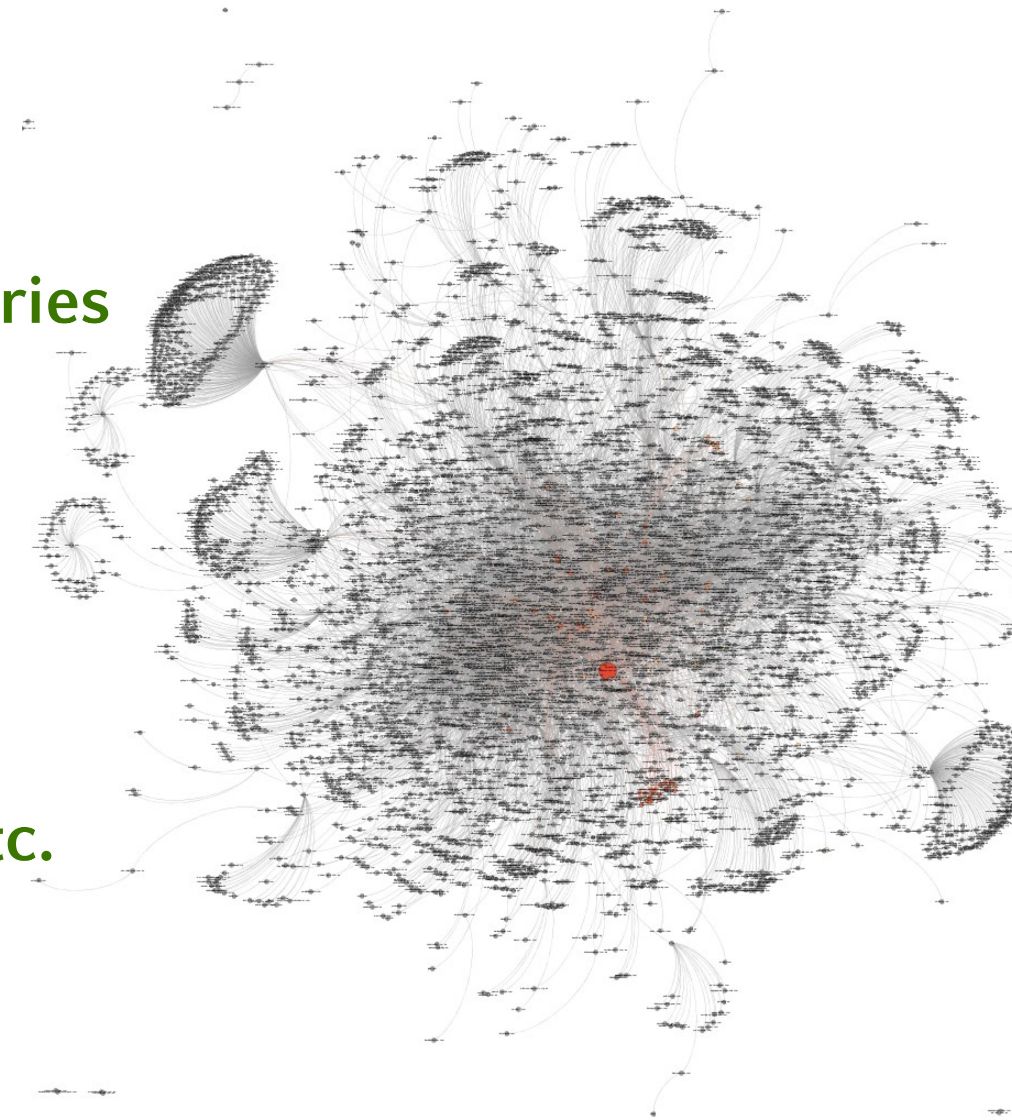
General Question

How to organise the web?

<u>First try:</u>
Human Curated Web Directories
Yahoo, DMOZ, LookSmart



General Question



How to organise the web?

First try:
Human Curated Web Directories

Second try:
Web Search

But: Web is huge,
full of untrusted documents,
random things, web spam, etc.

Web Search: Challenges

1) Web contains many sources of information.
   → Who to trust?

   Idea: Trustworthy pages may point to each other

2) What is the "best" answer to a certain query?
   → How to rank results?

   No single right answer.

Early Search Engines: Crawl the web, list terms, create inverted index

http://www.example.org|

Headline

This text contains words. Words are important. Many words appear in this text.

Early Search Engines: Crawl the web, list terms, create inverted index

http://www.example.org|

Headline

This text contains words. Words are important. Many words appear in this text.

appear	example.org (1)
are	example.org (1)
contains	example.org (1)
headline	example.org (1)
important	example.org (1)
in	example.org (1)
many	example.org (1)
text	example.org (2)
this	example.org (2)
words	example.org (3)

Problem:
Term Spam

Web Search: Ranking Results

Not all web pages are equally “important”

www.nytimes.com (The New York Times) vs. www.thetimesonline.com (The Times of Northwest Indiana, Munster, IN)

Web Search: Ranking Results

Not all web pages are equally "important"

www.nytimes.com (The New York Times) vs. www.thetimesonline.com (The Times of Northwest Indiana, Munster, IN)

in-links: ~13.600.000 vs. in-links: 5.960

→ There is a large diversity in the web-graph node connectivity.

IDEA: rank pages by their link structure!

Page Rank: “Flow” Formulation

Idea: links as votes

Page is more important if it has more links

In-links? Out-links?

Page Rank: "Flow" Formulation

Idea: links as votes

Page is more important if it has more in-links

Think of in-links as votes.

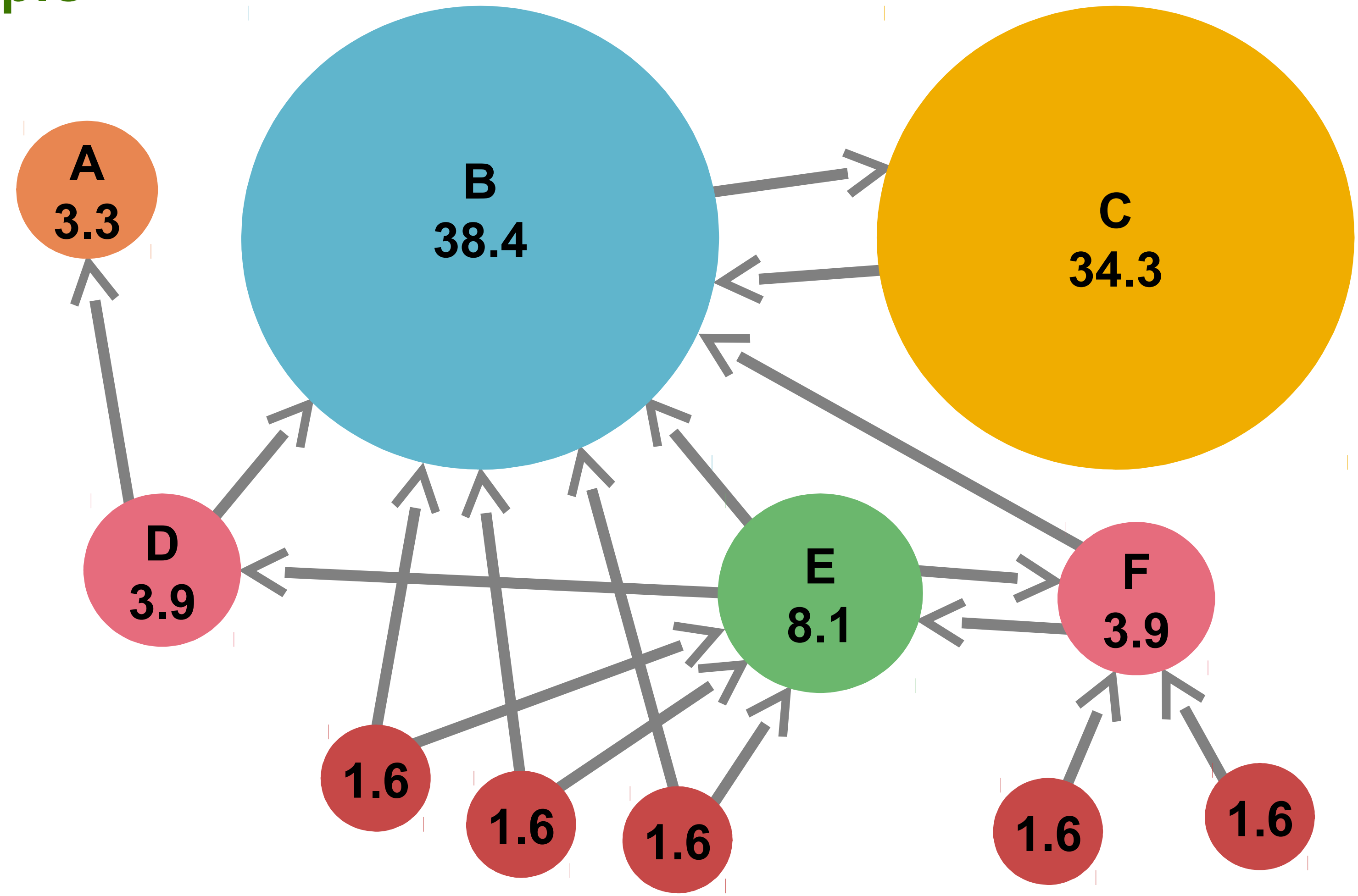
Are all in-links equal?

Links from important pages count more

=> Recursive Definition!

Page Rank: "Flow" Formulation

Example



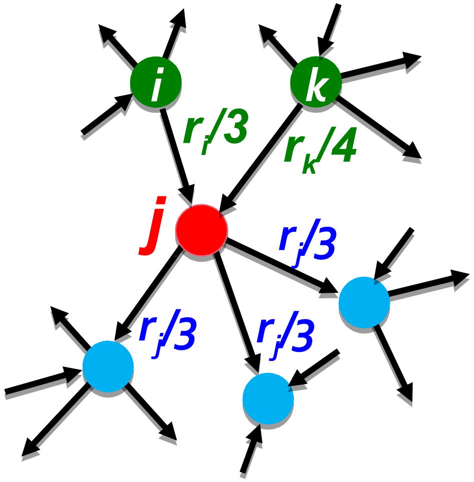
Simple Recursive Formulation

- Each link's vote is proportional to the importance of its source page

- If page j with importance r_j has n out-links, each link gets r_j / n votes

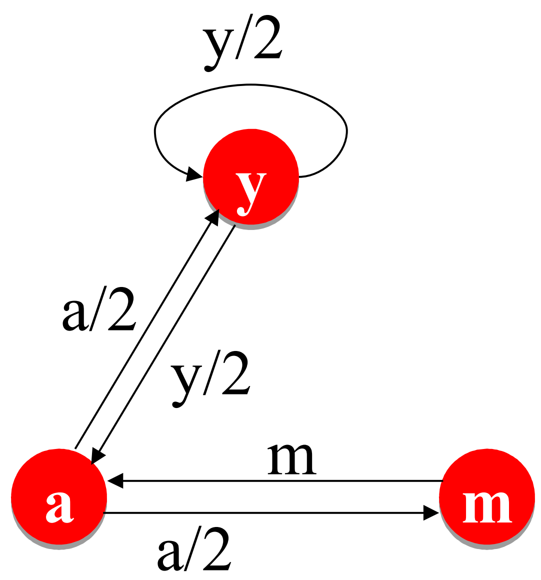
- Page j's own importance is the sum of the votes on its in-links

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



Page Rank: The "Flow" Model

- A "vote" from an important page is worth more
- A page is more important if it is pointed to by other important pages



Define a "rank" r_j for page j

(with d_i = out-degree of node i)

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

"Flow" equations:

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

Solving the Flow Equations

- 3 equations, 3 unknowns, no constants
  - No unique solution
  - All solutions equivalent modulo the scale factor
- Additional constraint forces uniqueness:
  - $r_y + r_a + r_m = 1$
  - Solution via Gaussian elimination $r_y = 2/5$, $r_a = 2/5$, $r_m = 1/5$
- Gaussian elimination method works for small examples, but we need a better method for large web-sized graphs
- We need a new formulation!

PageRank: Matrix Formulation

- **Stochastic adjacency matrix M**
  - Let page i has d_i out-links
  - If $i \rightarrow j$, then $M_{ji} = 1/d_i$, else $M_{ji} = 0$
  - M is a column stochastic matrix: columns sum to 1

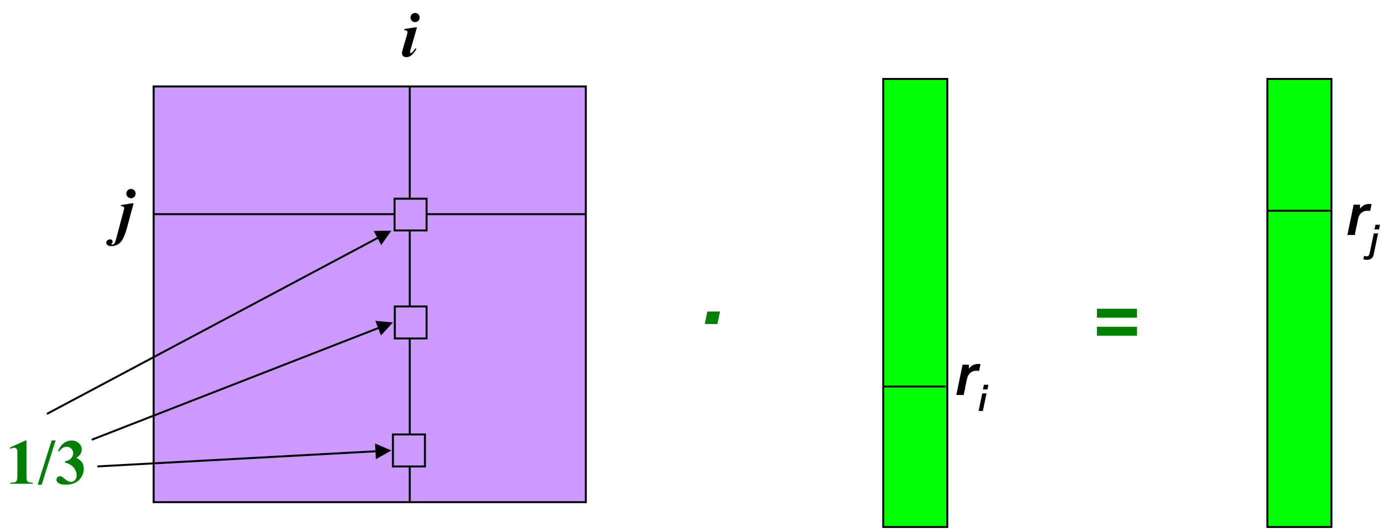
- **Rank vector r: vector with an entry per page**
  - r_i is the importance score of page i
  - $\Sigma_i\, r_i = 1$

- **The flow equations can be written**

$$r = M \cdot r$$

Example

- **Remember the flow equation:** $$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$
- **Flow equation in matrix form:** $M \cdot r = r$
  - **Suppose page i links to 3 pages, including j:**



$$M \cdot r = r$$

Eigenvector Formulation

- **The flow equations can be written as**

$$r = M \cdot r$$

- **So the rank vector *r* is an *eigenvector* of the stochastic web matrix *M***
  - In fact, its first or principal *eigenvector* with corresponding *eigenvalue* 1
  - Largest *eigenvalue* of *M* is 1 since *M* is column stochastic (with non-negative entries)
  - We know *r* is unit length and each column of *M* sums to 1, so $M \cdot r \leq 1$

Note:
x is an eigenvector with corresponding eigenvalue λ if:

$$Ax = \lambda x$$

- **We can now efficiently solve for *r*!**
  Power Iteration

Power Iteration

- **Power Iteration is an eigenvalue algorithm**
  - Also known as Von Mises iteration
  - Given a matrix A, P.I. returns a value λ and a nonzero vector v, such that $Av = \lambda v$

- **Will find only the dominant eigenvector (the vector corresponding to the largest eigenvalue)**

$$r^{(1)} = M \cdot r^{(0)}$$

$$r^{(2)} = M \cdot r^{(1)} = M\,(\,M \cdot r^{(0)}\,) = M^2 \cdot r^{(0)}$$

$$r^{(3)} = M \cdot r^{(2)} = M\,(\,M^2 \cdot r^{(0)}\,) = M^3 \cdot r^{(0)}$$

Power Iteration Method

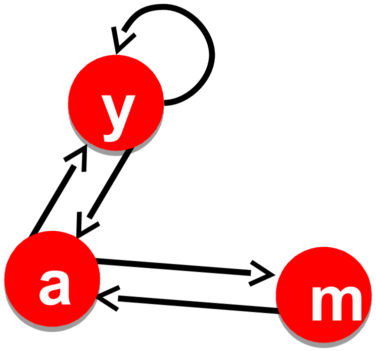
- Given a web graph with n nodes, where the nodes are pages and the edges are hyperlinks

- Power iteration: a simple iterative scheme
  - Suppose there are N web pages
  - Initialize: $r^{(0)} = [1/N, \ldots, 1/N]^T$
  - Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
  - Stop when: $| r^{(t+1)} - r^{(t)} |_1 < \varepsilon$

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

PageRank with Power Iteration

- **Power Iteration:**
  - Set $r_j = 1/N$
  - 1: $r'_j = \sum_{i \to j} r_i / d_i$
  - 2: $r = r'$
  - Goto 1



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

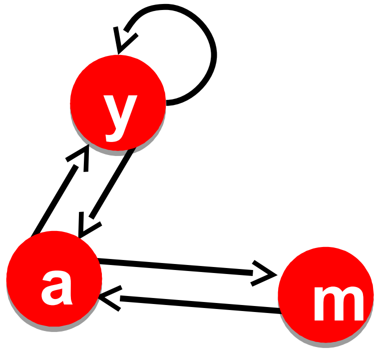
$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

PageRank with Power Iteration

- **Power Iteration:**
  - Set $r_j = 1/N$
  - 1: $r'_j = \sum_{i \to j} r_i / d_i$
  - 2: $r = r'$
  - Goto 1



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

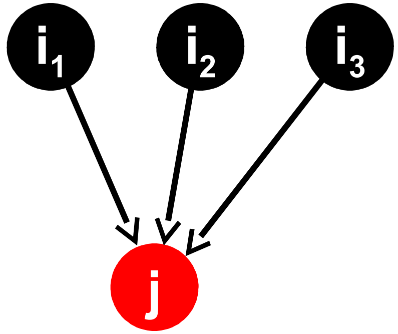
$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

- **Example:**

r_y		1/3	1/3	5/12	9/24		6/15
r_a	=	1/3	3/6	1/3	11/24	…	6/15
r_m		1/3	1/6	3/12	1/6		3/15

Random Walk Interpretation

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

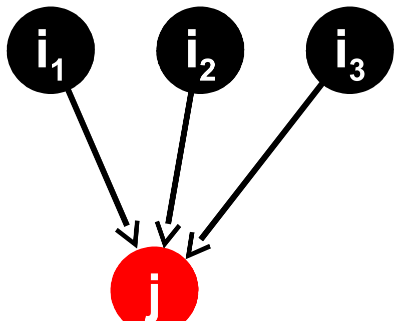


$$r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d_{out}(i)}}$$

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

Random Walk Interpretation

- **Where is surfer at time t + 1?**
  - Follows a link uniformly at random
    $p(t+1) = M \cdot p(t)$



$$r_j = \sum_{i \to j} \frac{r_i}{d_{out}(i)}$$

- **Suppose the random walk reaches a state**
  $p(t+1) = M \cdot p(t) = p(t)$
  then p (t) is stationary distribution of a random walk

- **Our original rank vector r satisfies $r = M \cdot r$**
  - So, r is a stationary distribution for a random walk

Existence and Uniqueness

A central result from the theory of random walks (a.k.a. Markov processes):

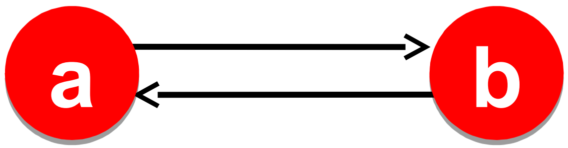
For graphs that satisfy **certain conditions**, the **stationary distribution is unique** and eventually will be reached no matter what the initial probability distribution at time t = 0.

PageRank in real life

$$r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{\mathrm{d_i}} \qquad r = Mr$$

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

Does this converge?

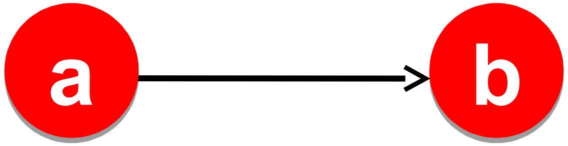


$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

- **Example:**

r_a	1	0	1	0	1	0	1	0	…
r_b	0	1	0	1	0	1	0	1	

Does it converge to what we want?



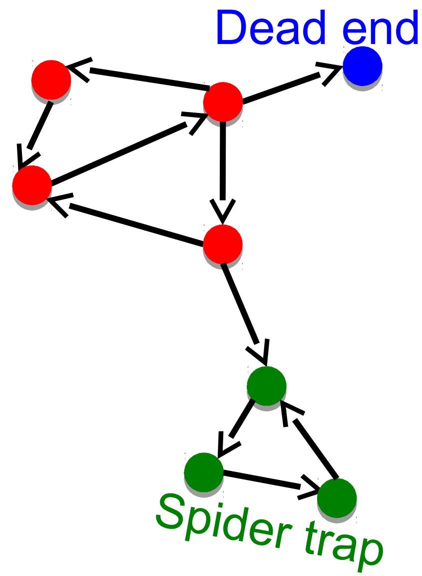
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_\mathrm{i}}$$

- **Example:**

r_a	1	0	0	0	0	0	0	…
r_b	0	1	0	0	0	0	0	

PageRank: Problems

2 Problems:



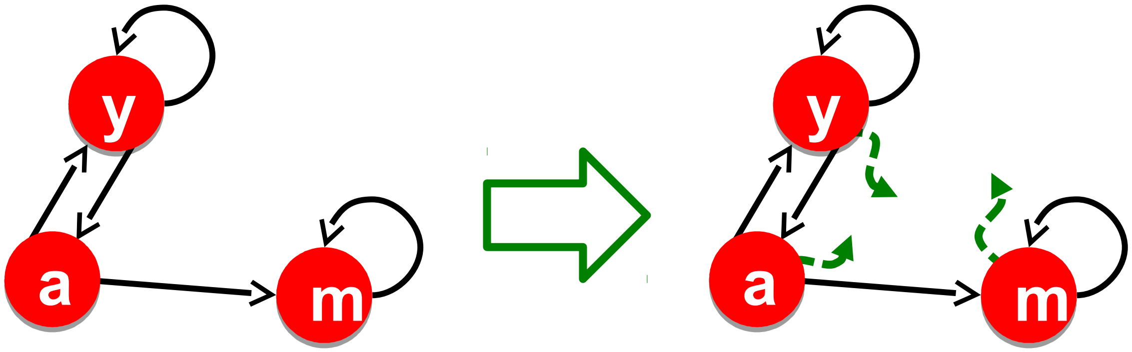
- **Some pages are dead ends (have no out-links)**
  - Random walk has "nowhere to go" to
  - Such pages cause "leak" of importance
- **Spider traps (all out-links are within a group)**
  - Random walk gets "stuck" in a trap
  - Eventually spider trap absorbs all importance

The Google solution for spider traps: *Teleports*

At each time step, the random surfer has two options:

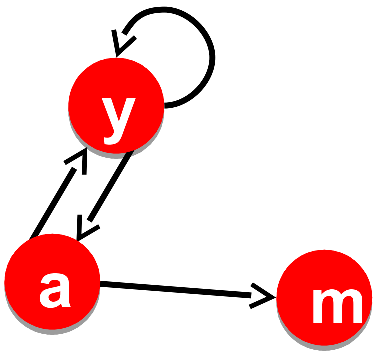
- With probability ß, follow a link at random
- With probability 1 – ß, jump to some random page
- Common values for ß range between 0.8 and 0.9

Surfer will teleport out of spider trap within a few time steps



Dead Ends

Dead ends cause the page importance to leak out, because the adjacency matrix is non-stochastic.

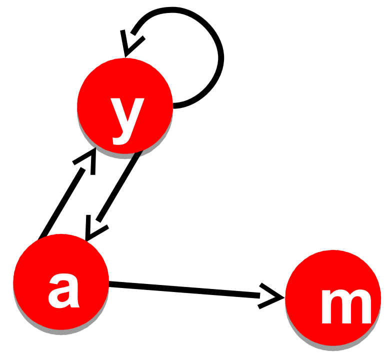


	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

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

Dead Ends: Solution

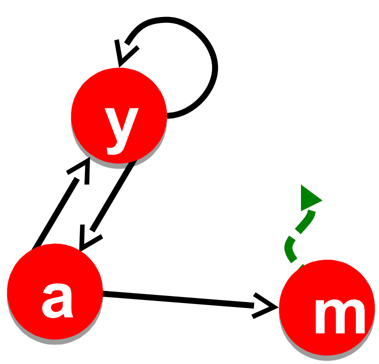
Dead ends cause the page importance to leak out, because the adjacency matrix is non-stochastic.



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

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

Solution: Always teleport!
Adjust matrix accordingly:



	y	a	m
y	½	½	⅓
a	½	0	⅓
m	0	½	⅓

The final version of the Google PageRank: *[Brin-Page 98]*

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

(This formulation assumes M has no dead ends.
M can either be preprocessed to remove all dead ends
or with explicit teleports to random links from dead ends.)

The Google Matrix

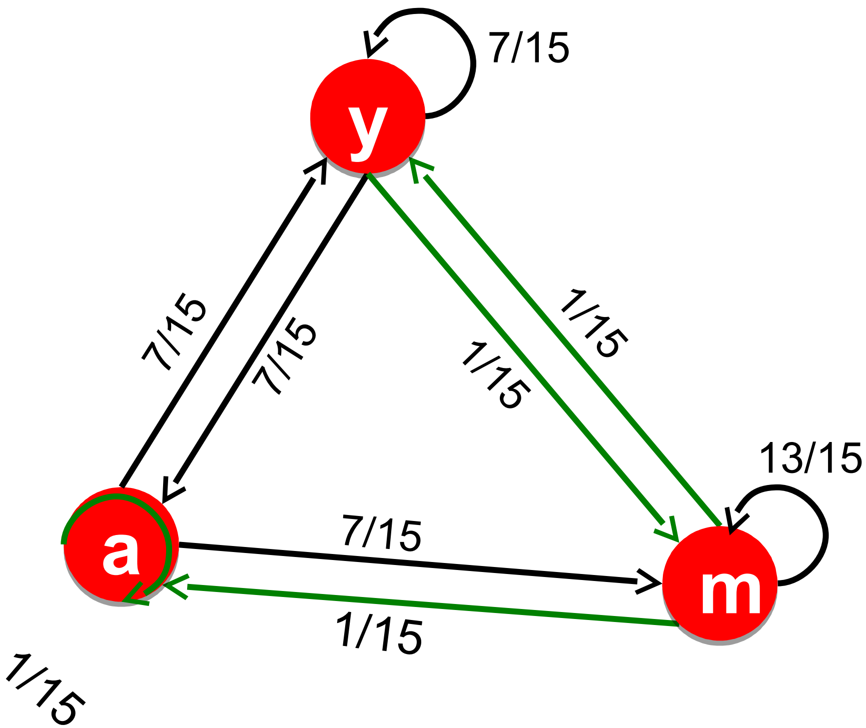
Google matrix A combines the adjacency matrix M with the random teleports by a factor ß.

(With ß = 0.8 for this example)

$$ß \underbrace{\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}}_{\mathbf{M}} + 1 - ß \underbrace{\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}}_{[1/N]_{NxN}}$$

$$\begin{array}{c} y \\ a \\ m \end{array} \underbrace{\begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}}_{\mathbf{A}}$$

The Google Matrix



$$
0.8 \underbrace{\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}}_{\mathbf{M}} + 0.2 \underbrace{\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}}_{[1/N]_{NxN}}
$$

$$
\mathbf{A} = \begin{array}{c} y \\ a \\ m \end{array} \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}
$$

y		1/3	0.33	0.24	0.26		7/33
a	=	1/3	0.20	0.20	0.18	…	5/33
m		1/3	0.46	0.52	0.56		21/33

Some Problems with PageRank

- **Measures generic popularity of a page**
  - Biased against topic-specific authorities
  - Solution: Topic-specific PageRank
- **Uses only one measure of importance**
  - Other models exist
  - Solution: e.g., Hubs and Authorities
- **Susceptible to Link Spam**
  - Evolved from term spam (see: older search engines)
  - Artificial link topographies created to boost page rank
  - Solution: TrustRank

Topic-specific PageRank

- Instead of generic popularity, can we measure popularity within a certain topic?
- Goal: evaluate web pages not only according to their popularity, but by how close they are to a particular topic, e.g., "sports" or "history"
- Allows search queries to be answered based on user interest
  - Example: Query "Trojan" may yield different results depending on whether user is interested in sports, history, computer security, …

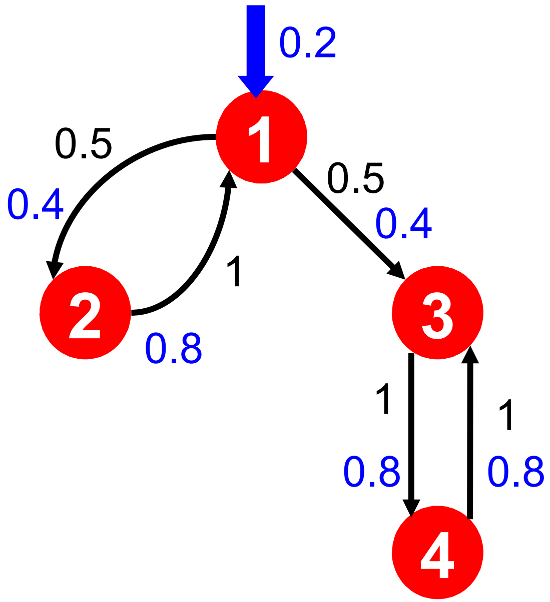
Topic-specific PageRank

- **Modification in random walk behaviour (teleports)**

- **Teleport has probability to go to:**
  - Standard PageRank: Any page with equal probability
    To avoid dead ends and spider-traps
  - Topic-specific PageRank: A topic specific set of "relevant" pages (teleport set)

- **Idea: Bias the random walk**
  - When walker teleport, they pick a page from set S
  - S contains only pages that are relevant to the topic, e.g., from Open Directory (DMOZ) pages for given topic
  - For each teleport set S, we get a different vector r_S

Example: Topic-specific PageRank



Suppose $S = \{1\}$, $\beta = 0.8$

Node	Iteration				
	0	**1**	**2**	**…**	**stable**
1	0.25	0.4	0.28		0.294
2	0.25	0.1	0.16		0.118
3	0.25	0.3	0.32		0.327
4	0.25	0.2	0.24		0.261

S={1}, β=0.90:
r=[0.17, 0.07, 0.40, 0.36]
S={1} , β=0.8:
r=[0.29, 0.11, 0.32, 0.26]
S={1}, β=0.70:
r=[0.39, 0.14, 0.27, 0.19]

S={1,2,3,4}, β=0.8:
r=[0.13, 0.10, 0.39, 0.36]
S={1,2,3} , β=0.8:
r=[0.17, 0.13, 0.38, 0.30]
S={1,2} , β=0.8:
r=[0.26, 0.20, 0.29, 0.23]
S={1} , β=0.8:
r=[0.29, 0.11, 0.32, 0.26]

Topic vector S

- **Create different PageRanks for different topics**
  - The 16 DMOZ top-level categories
    art, business, sports, …

- **Which topic ranking to use?**
  - User can pick from a menu
  - Classify query into a topic
  - Use context of query:
    e.g., query is launched from website about certain topic, or history of queries
  - User context, e.g., bookmarks, ...

PageRank Summary

- **"Normal" PageRank**
  - Teleports uniformly at random to any node
  - All nodes have the same landing probability
    S = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]

- **Topic-specific PageRank, also known as Personalized PageRank**
  - Teleports to a topic specific set of pages
  - Nodes can have different landing probabilities
    S = [0.1, 0.0, 0.2, 0.0, 0.0, 0.0, 0.5, 0.0, 0.2, 0.0]

- **Random walk with restarts**
  - Topic-specific with teleports to always the same node
    S = [0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

Link Spam

- **Spamming:**
  Any deliberate action with the intent to boost a web page's position in search engine results incommensurate with page's actual relevance

- **Spam:**
  Query results that are the result of spamming

  → very broad definition

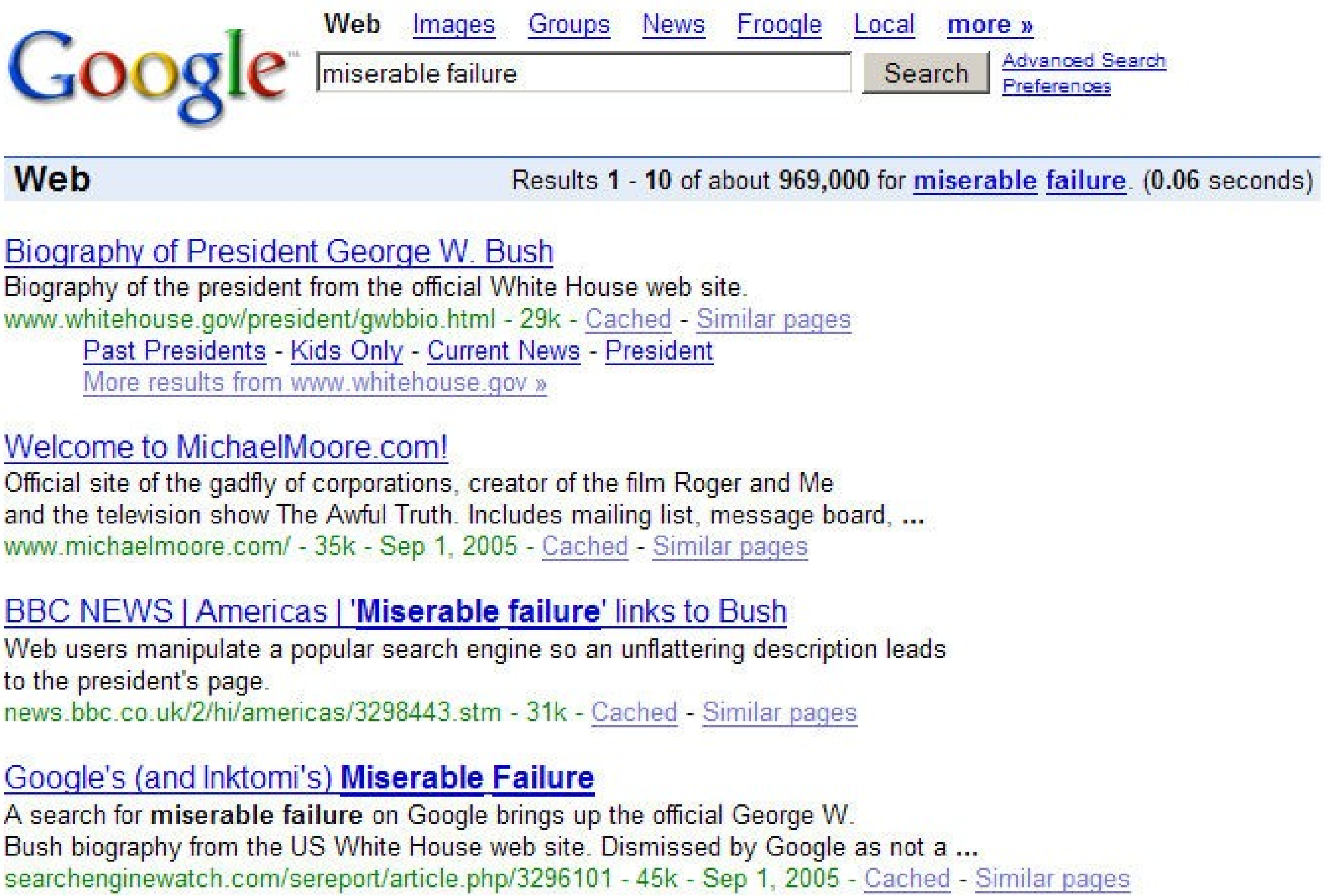
- Approximately 10% – 15% of web pages are spam

Link Spam

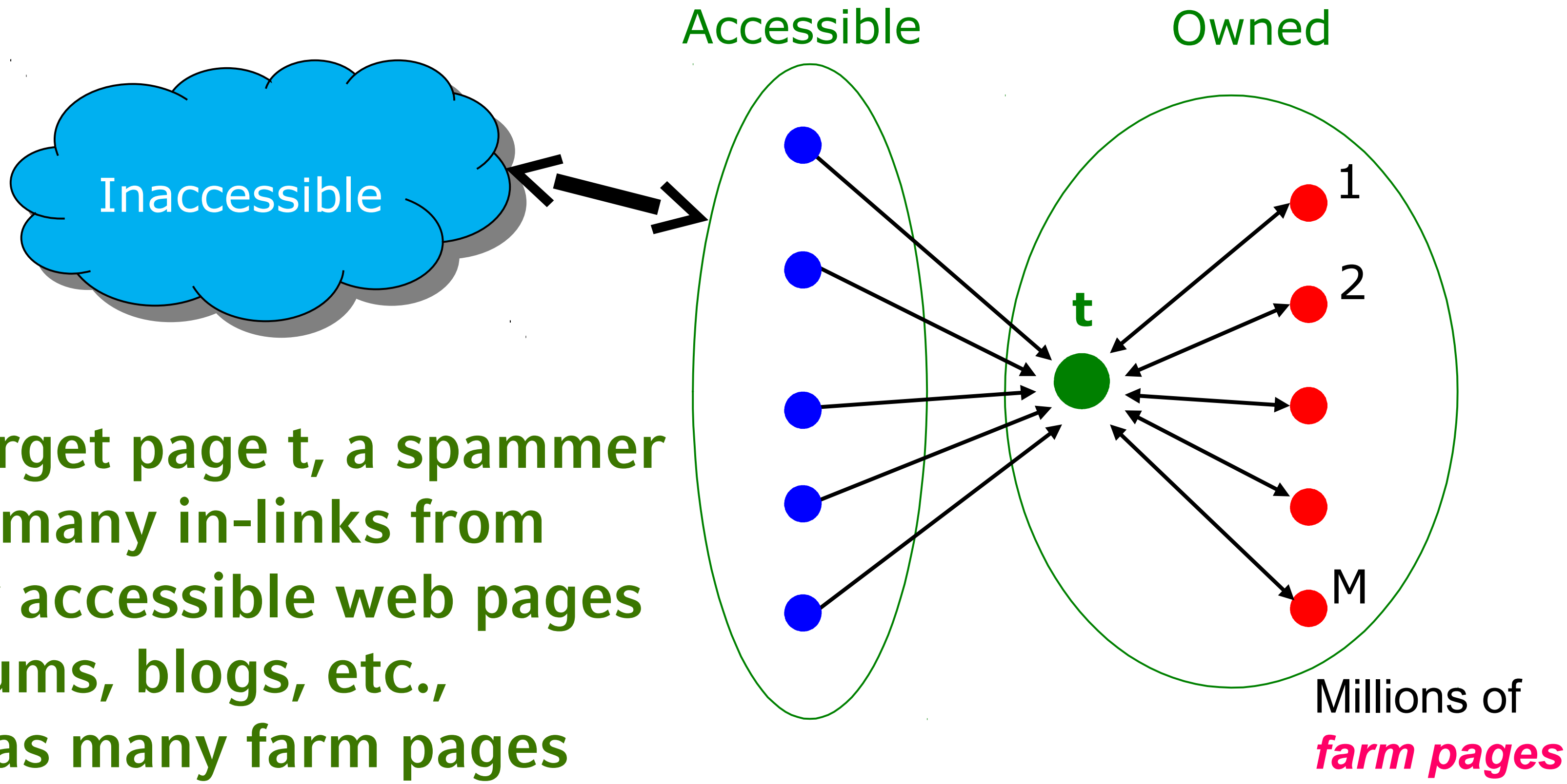
- **Early spamming techniques flooded web pages with unfitting words to exploit search engines**
  - Example: Web page for T-Shirts includes the word "movie" over and over again
  - "Term spam"

- **As Google became more dominant, spam farms tried to target PageRank to a single page by placing many contextual links on other pages**
  - "Link Spam" or "Google Bomb"

2003 George W. Bush Google Bomb



Link Farms



For a target page t, a spammer creates many in-links from publicly accessible web pages like forums, blogs, etc., as well as many farm pages on own infrastructure to create a closely connected clique.

Combating Spam

- **Combating Term Spam:**
  - Analyze text using statistical methods
  - Similar to email spam filtering
  - Detecting duplicate pages

- **Combating Link Spam:**
  - Detection and blacklisting of structures that look like spam farms
  - Leads to another war: hiding and detecting

  - TrustRank = topic-specific PageRank with teleport to a set of trusted pages, e.g., .edu domains or similar

TrustRank

- **Alternative model for TrustRank: Trust Propagation**

  Initial seed set of trusted pages (evaluated by hand)

- **Set trust tp of each trusted page p to 1**
  - For each out-link from p, a portion of the trust is passed on to target page q

- **Trust is additive**
  - Trust of q is sum of all trust conferred by its in-links

- **If trust is below a threshold, page is flagged as spam**

- Exam on the 5th of February, 2016, 14.00 to 16.00

- If you wish to attend, please register!

http://uniworx.ifi.lmu.de/