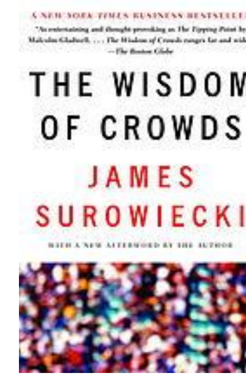
Chapter 14: Link Analysis

We didn't know exactly what I was going to do with it,
but no one was really looking at the links on the Web.
In computer science, there's a lot of big graphs.

-- Larry Page

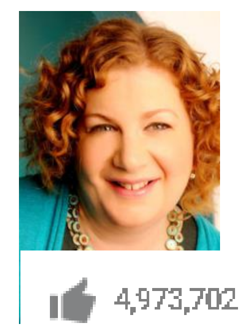
The many are smarter than the few.

-- James Surowiecki



Like, like, like – my confidence grows with every click.

-- Keren David



Money isn't everything ... but it ranks right up there with oxygen.

-- Rita Davenport

Outline

14.1 PageRank for Authority Ranking

14.2 Topic-Sensitive, Personalized & Trust Rank

14.3 HITS for Authority and Hub Ranking

14.4 Extensions for Social & Behavioral Ranking

following Büttcher/Clarke/Cormack Chapter 15
and/or Manning/Raghavan/Schuetze Chapter 21

Google‘s PageRank [Brin & Page 1998]

<u>Idea:</u> links are endorsements & increase page authority, authority higher if links come from high-authority pages

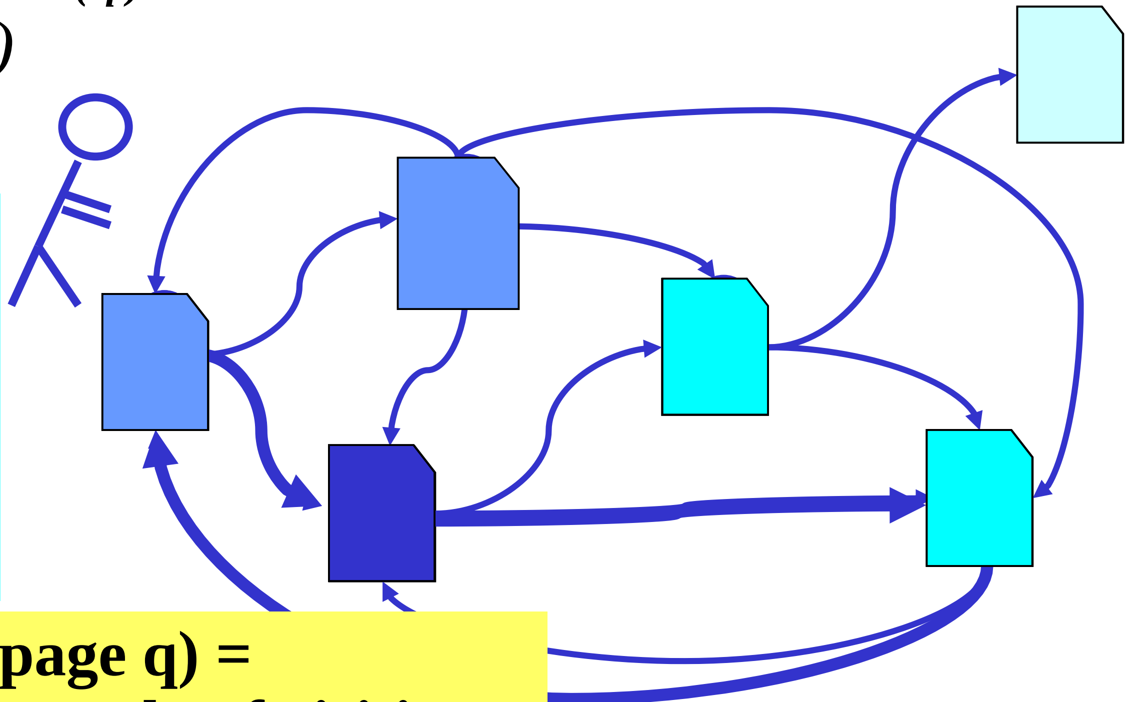
$$PR(q) = \varepsilon \cdot j(q) + (1-\varepsilon) \cdot \sum_{p \in IN(q)} PR(p) \cdot t(p,q)$$

Wisdom of Crowds

with $t(p,q) = 1/outdegree(p)$

and $j(q) = 1/N$

Extensions with
- **weighted links and jumps**
- **trust/spam scores**
- **personalized preferences**
- **graph derived from queries & clicks**



Authority (page q) = stationary prob. of visiting q

random walk: uniformly random choice of links + random jumps

Role of PageRank in Query Result Ranking

- PageRank (PR) is a **static (query-independent) measure** of a page's or site's authority/prestige/importance

- Models for query result ranking **combine** PR with query-dependent content score (and freshness etc.):
  - linear combination of PR and score by LM, BM25, …
  - PR is viewed as doc prior in LM
  - PR is a feature in Learning-to-Rank

Simplified PageRank

given: directed Web graph G=(V,E) with |V|=n and
adjacency matrix E: $E_{ij} = 1$ if $(i,j) \in E$, 0 otherwise

random-surfer page-visiting probability after i +1 steps:

$$p^{(i+1)}(y) = \sum_{x=1..n} C_{yx}\, p^{(i)}(x)$$

with conductance matrix C:
$C_{yx} = E_{xy} / out(x)$

$$p^{(i+1)} = C\, p^{(i)}$$

finding solution of fixpoint equation p = Cp suggests
power iteration:
initialization: $p^{(0)}(y) = 1/n$ for all y
repeat until convergence (L_1 or L_∞ of diff of $p^{(i)}$ and $p^{(i+1)}$ < threshold)
$p^{(i+1)} := C\, p^{(i)}$

PageRank as Principal Eigenvector of Stochastic Matrix

A **stochastic matrix** is an $n \times n$ matrix M
with row sum $\Sigma_{j=1..n} M_{ij} = 1$ for each row i

Random surfer follows a stochastic matrix

Theorem (special case of Perron-Frobenius Theorem):
For every stochastic matrix M
all Eigenvalues λ have the property $|\lambda| \leq 1$
and there is an Eigenvector x with Eigenvalue 1 s.t. $x \geq 0$ and $||x||_1 = 1$

Suggests power iteration $x^{(i+1)} = M^T x^{(i)}$

But: real Web graph
has sinks, may be periodic, is not strongly connected

Dead Ends and Teleport

Web graph has sinks (dead ends, dangling nodes)
Random surfer can‘t continue there

Solution 1: remove sinks from Web graph

Solution 2: introduce random jumps (teleportation)

if node y is sink then jump to randomly chosen node
else with prob. α choose random neighbor by outgoing edge
with prob. $1-\alpha$ jump to randomly chosen node

$\rightarrow$ fixpoint equation $p = C\,p$

generalized into: $p = \alpha\,C\,p + (1-\alpha)\,r$

with $n \times 1$ teleport vector r
with $r_y = 1/n$ for all y
and $0 < \alpha < 1$
(typically $0.15 < 1-\alpha < 0.25$)

Power Iteration for General PageRank

power iteration (Jacobi method):

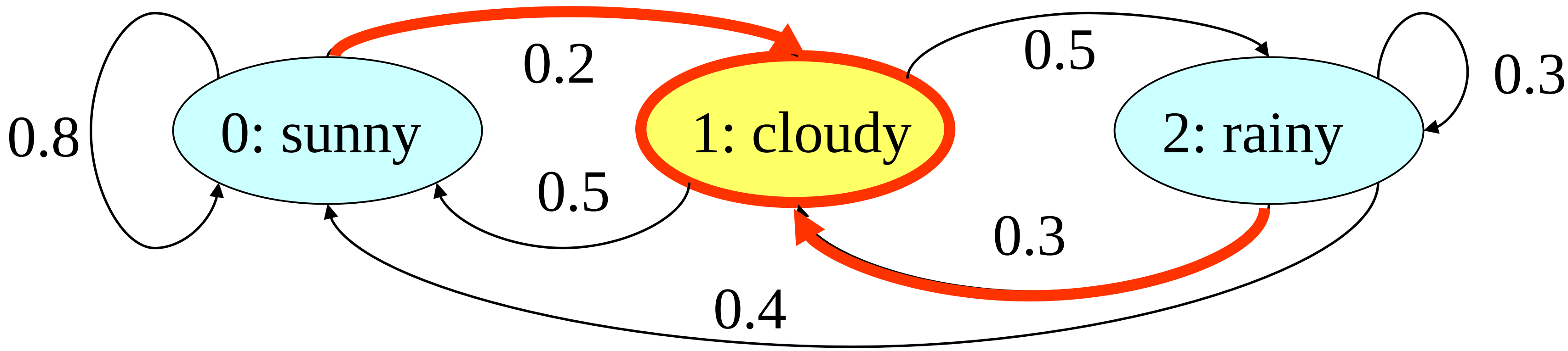
initialization: $p^{(0)}(y) = 1/n$ for all y

repeat until convergence (L_1 or L_∞ of diff of $p^{(i)}$ and $p^{(i+1)}$ < threshold)

$$\mathbf{p}^{(i+1)} := \alpha\, \mathbf{C}\, \mathbf{p}^{(i)} + (1-\alpha)\, \mathbf{r}$$

- scalable for huge graphs/matrices
- convergence and uniqueness of solution guaranteed
- implementation based on adjacency lists for nodes y
- termination criterion based on stabilizing ranks of top authorities
- convergence typically reached after ca. 50 iterations
- convergence rate proven to be: $|\lambda_2 / \lambda_1| = \alpha$ with second-largest eigenvalue λ_2 [Havelivala/Kamvar 2002]

Markov Chains (MC) in a Nutshell



p0 = 0.8 p0 + 0.5 p1 + 0.4 p2
p1 = 0.2 p0 + 0.3 p2
p2 = 0.5 p1 + 0.3 p2
p0 + p1 + p2 = 1

$\Rightarrow$ p0 $\approx$ 0.657, p1 = 0.2, p2 $\approx$ 0.143

state set: finite or infinite

time: discrete or continuous

state transition prob's: p_{ij}

state prob's in step t: $p_i^{(t)} = P[S(t)=i]$

Markov property: P[S(t)=i | S(0), ..., S(t-1)] = P[S(t)=i | S(t-1)]

interested in **stationary state probabilities**:

$$p_j := \lim_{t\to\infty} p_j^{(t)} = \lim_{t\to\infty} \sum_k p_k^{(t-1)} p_{kj}$$

$$p_j = \sum_k p_k p_{kj} \qquad \sum_j p_j = 1$$

exist & unique for irreducible, aperiodic, finite MC (**ergodic MC**)

Digression: Markov Chains

A **stochastic process** is a family of random variables $\{X(t) \mid t \in T\}$.
T is called parameter space, and the domain M of X(t) is called state space. T and M can be discrete or continuous.

A stochastic process is called **Markov process** if for every choice of $t_1, ..., t_{n+1}$ from the parameter space and every choice of $x_1, ..., x_{n+1}$ from the state space the following holds:

$$P[\,X(t_{n+1}) = x_{n+1} \mid X(t_1) = x_1 \wedge X(t_2) = x_2 \wedge ... \wedge X(t_n) = x_n\,]$$
$$= P[\,X(t_{n+1}) = x_{n+1} \mid X(t_n) = x_n\,]$$

A Markov process with discrete state space is called **Markov chain**.
A canonical choice of the state space are the natural numbers.
Notation for Markov chains with discrete parameter space:
X_n rather than $X(t_n)$ with n = 0, 1, 2, ...

Properties of Markov Chains with Discrete Parameter Space (1)

The Markov chain Xn with discrete parameter space is

homogeneous if the transition probabilities $p_{ij} := P[X_{n+1} = j \mid X_n = i]$ are independent of n

irreducible if every state is reachable from every other state with positive probability:

$$\sum_{n=1}^{\infty} P[X_n = j \mid X_0 = i] > 0 \quad \text{for all i, j}$$

aperiodic if every state i has period 1, where the period of i is the gcd of all (recurrence) values n for which

$$P[X_n = i \wedge X_k \neq i \text{ for } k = 1, \ldots, n-1 \mid X_0 = i] > 0$$

Properties of Markov Chains with Discrete Parameter Space (2)

The Markov chain Xn with discrete parameter space is

positive recurrent if for every state i the recurrence probability is 1 and the mean recurrence time is finite:

$$\sum_{n=1}^{\infty} P[X_n = i \wedge X_k \neq i \text{ for } k = 1, \ldots, n-1 \mid X_0 = i] = 1$$

$$\sum_{n=1}^{\infty} n\, P[X_n = i \wedge X_k \neq i \text{ for } k = 1, \ldots, n-1 \mid X_0 = i] < \infty$$

ergodic if it is homogeneous, irreducible, aperiodic, and positive recurrent.

Results on Markov Chains with Discrete Parameter Space (1)

For the **n-step transition probabilities**

$p_{ij}^{(n)} := P[\,X_n = j \mid X_0 = i\,]$ the following holds:

$$p_{ij}^{(n)} = \sum_k p_{ik}^{(n-1)} p_{kj} \quad \text{with} \quad p_{ij}^{(1)} := p_{ik}$$

$$= \sum_k p_{ik}^{(n-l)} p_{kj}^{(l)} \quad for\, 1 \le l \le n-1$$

in matrix notation: $P^{(n)} = P^n$

For the **state probabilities after n steps**

$\pi_j^{(n)} := P[\,X_n = j\,]$ the following holds:

$$\pi_j^{(n)} = \sum_i \pi_i^{(0)} p_{ij}^{(n)}$$ with initial state probabilities $\pi_i^{(0)}$

in matrix notation: $\Pi^{(n)} = \Pi^{(0)} P^{(n)}$

(Chapman-Kolmogorov equation)

Results on Markov Chains with Discrete Parameter Space (2)

Theorem: Every homogeneous, irreducible, aperiodic Markov chain with a finite number of states is ergodic.

For every ergodic Markov chain there exist **stationary state probabilities**

$$\pi_j := \lim_{n \to \infty} \pi_j^{(n)}$$

These are independent of $\Pi^{(0)}$
and are the solutions of the following system of linear equations:

$$\pi_j = \sum_i \pi_i \, p_{ij} \quad \text{for all } j$$

(balance equations)

$$\sum_j \pi_j = 1$$

in matrix notation:

(with $1 \times n$ row vector Π)

$$\Pi = \Pi P$$

$$\Pi \vec{1} = 1$$

Page Rank as a Markov Chain Model

Model a **random walk** of a Web surfer as follows:

- follow outgoing hyperlinks with uniform probabilities
- perform „random jump“ with probability $1-\alpha$

$\rightarrow$ ergodic Markov chain

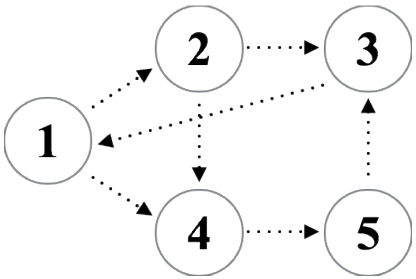
PageRank of a page is its **stationary visiting probability**

(uniquely determined and independent of starting condition)

Further generalizations have been studied

(e.g. random walk with back button etc.)

Page Rank as a Markov Chain Model: Example



$$G = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad C = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 1/2 & 1/2 & 0 \\ 1/1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/1 \\ 0 & 0 & 1/1 & 0 & 0 \end{bmatrix}$$

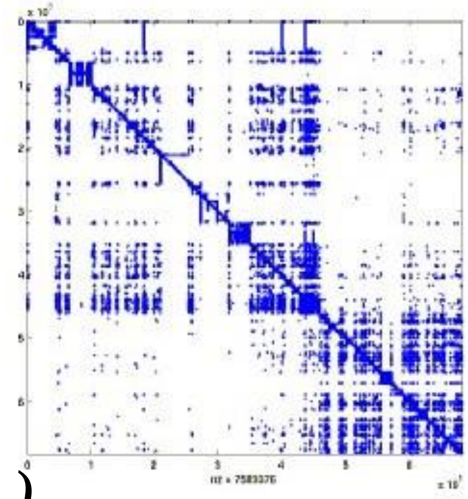
with ε=0.15 $\mathbf{P} = \begin{bmatrix} 0.030 & 0.455 & 0.030 & 0.455 & 0.030 \\ 0.030 & 0.030 & 0.455 & 0.455 & 0.030 \\ 0.880 & 0.030 & 0.030 & 0.030 & 0.030 \\ 0.030 & 0.030 & 0.030 & 0.030 & 0.880 \\ 0.030 & 0.030 & 0.880 & 0.030 & 0.030 \end{bmatrix}$

approx. solution of Pπ= π

$$\boldsymbol{\pi} = \begin{bmatrix} 0.24079 & 0.13234 & 0.24799 & 0.18858 & 0.19029 \end{bmatrix}$$

Efficiency of PageRank Computation

[Kamvar/Haveliwala/Manning/Golub 2003]



(b) Stanford/Berkeley

Exploit **block structure of the link graph**:

1) partitition link graph by domains (entire web sites)

2) compute **local PR vector** of pages within each block → LPR(i) for page i

3) compute **block rank** of each block:

a) block link graph B with $B_{IJ} = \sum_{i \in I, j \in J} C^T{}_{ij} \cdot LPR(i)$

b) run PR computation on B, yielding BR(I) for block I

4) Approximate **global PR vector** using LPR and BR:

a) set $x_j^{(0)}$:= LPR(j) · BR(J) where J is the block that contains j

b) run PR computation on A

speeds up convergence by factor of 2 in good "block cases"
unclear how effective it is in general

Efficiency of Storing PageRank Vectors

[T. Haveliwala, Int. Conf. On Internet Computing 2003]

Memory-efficient encoding of PR vectors
(especially important for large number of PPR vectors)

<u>Key idea:</u>

- map real PR scores to n cells and encode cell no into ceil($\log_2$ n) bits
- approx. PR score of page i is the mean score of the cell that contains i
- should use non-uniform partitioning of score values to form cells

Possible encoding schemes:

- ***Equi-depth partitioning***: choose cell boundaries such that

  $$\sum_{i \in cell\ j} PR(i)$$ is the same for each cell

- ***Equi-width partitioning with log values***: first transform all PR values into log PR, then choose equi-width boundaries
- Cell no. could be variable-length encoded (e.g., using Huffman code)

Link-Based Similarity Search: SimRank

[G. Jeh, J. Widom: KDD 2002]

Idea: nodes p, q are similar if their in-neighbors are pairwise similar

$$sim(p,q) = \frac{1}{|In(p)||In(q)|} \sum_{x \in In(p)}) \sum_{y \in In(q} sim(x,y)$$

with sim(x,x)=1

Examples: 2 users and their friends or people they follow
2 actors and their co-actors or their movies
2 people and the books or food they like

Efficient computation [Fogaras et al. 2004]:

- compute RW fingerprint for each node p: ≈ P[reach node q]
- SimRank(p,q) ~ P[walks from p and q meet]
  → test on fingerprints (viewed as iid samples)

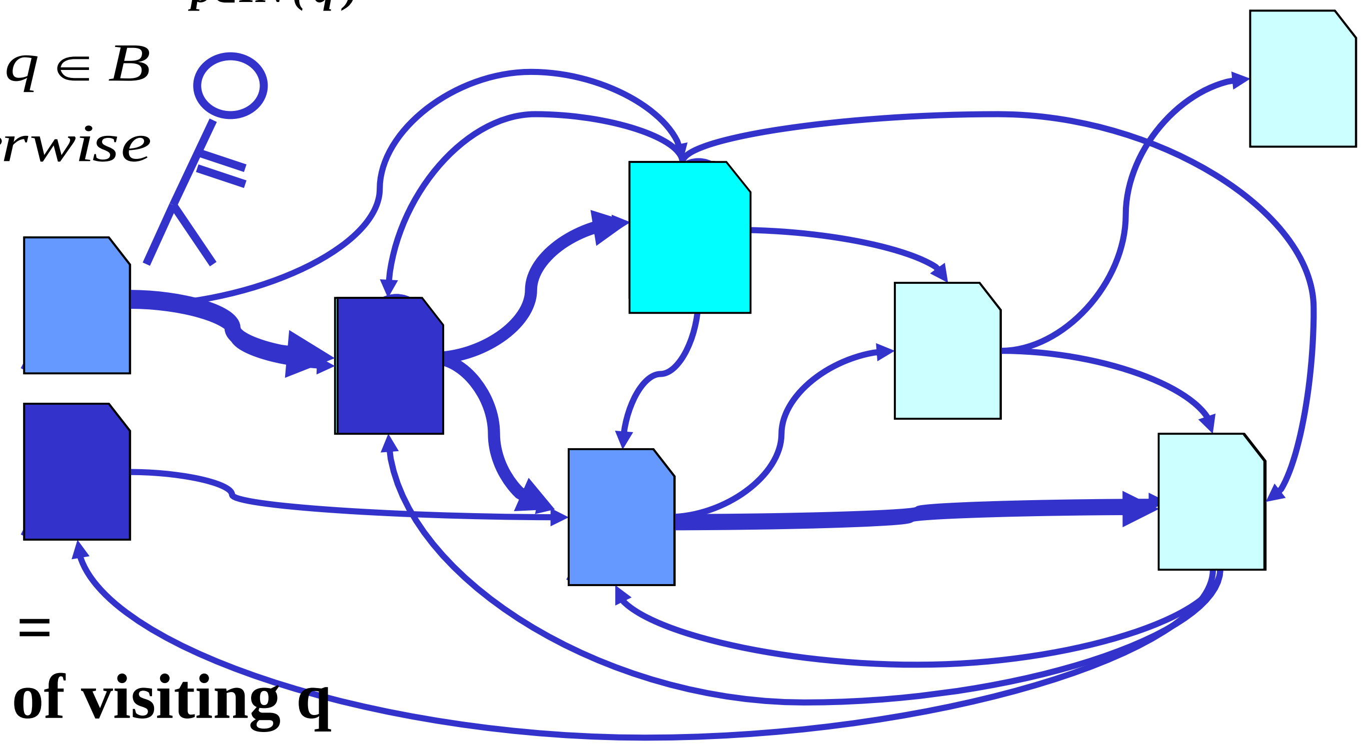
14.2 Topic-Specific & Personalized PageRank

<u>Idea:</u> random jumps favor pages of personal interest such as bookmarks, frequently&recently visited pages etc.

$$PR(q) = \varepsilon \cdot j(q) + (1-\varepsilon) \cdot \sum_{p \in IN(q)} PR(p) \cdot t(p,q)$$

with

$$j(q) = \begin{cases} 1/|B| & for\ \ q \in B \\ 0 & otherwise \end{cases}$$



Authority (page q) = stationary prob. of visiting q

random walk: uniformly random choice of links + biased jumps to personal favorites

Personalized PageRank

Goal: Efficient computation and efficient storage of user-specific **personalized PageRank vectors (PPR)**

PageRank equation: $p = \alpha\, C\, p + (1-\alpha)\, r$

Linearity Theorem:

Let r_1 and r_2 be personal preference vectors for random-jump targets, and let p_1 and p_2 denote the corresponding PPR vectors.
Then for all $\beta_1, \beta_2 \geq 0$ with $\beta_1 + \beta_2 = 1$ the following holds:

$$\beta_1 p_1 + \beta_2 p_2 = \alpha\, C\, (\beta_1 p_1 + \beta_2 p_2) + (1-\alpha)\, (\beta_1 r_1 + \beta_2 r_2)$$

Corollary:

For preference vector r with m non-zero components and base vectors e_k ($k=1..m$) with $(e_k)_i = 1$ for $i=k$, 0 for $i \neq k$, we obtain:

$$r = \sum_{k=1..m} \beta_k e_k \quad \text{with constants } \beta_1 \ldots \beta_m$$

and $p = \sum_{k=1..m} \beta_k p_k$ for PPR vector p with $p_k = \alpha\, C\, p_k + (1-\alpha)\, e_k$

for further optimizations see Jeh/Widom: WWW 2003

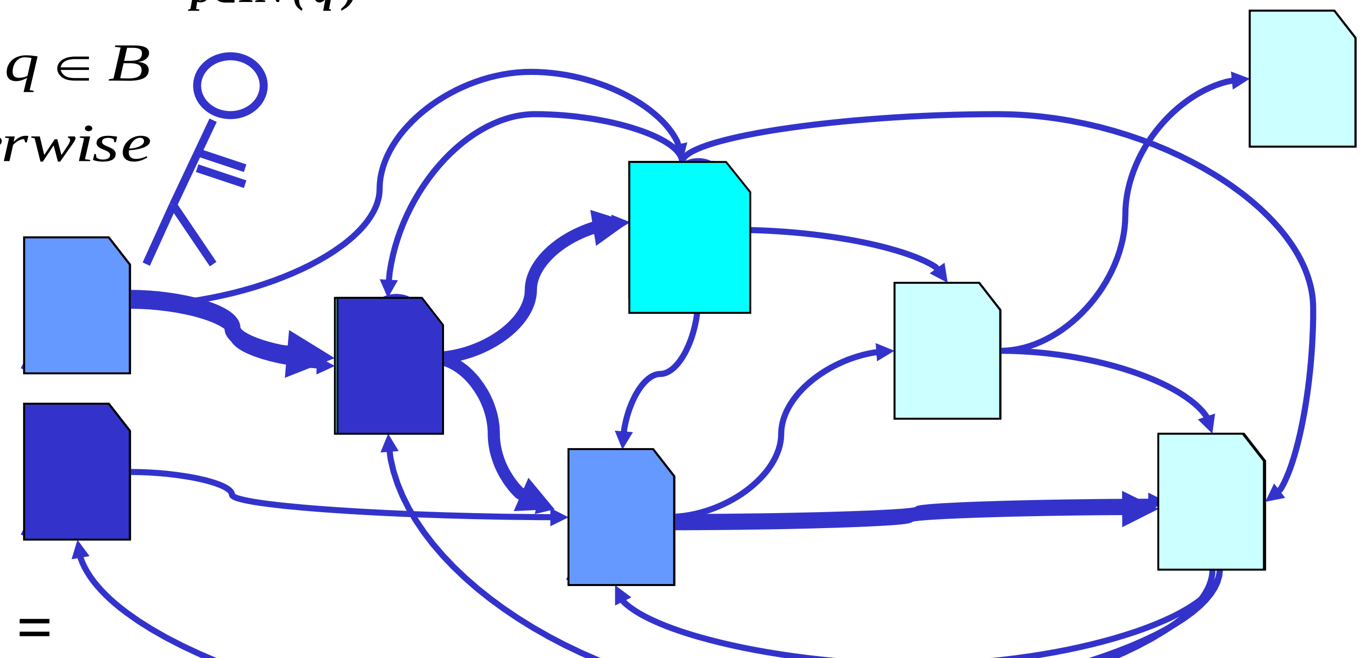
Spam Control: From PageRank to TrustRank

<u>Idea:</u> random jumps favor designated high-quality pages such as popular pages, trusted hubs, etc.

$$PR(q) = \varepsilon \cdot j(q) + (1-\varepsilon) \cdot \sum_{p \in IN(q)} PR(p) \cdot t(p,q)$$

with

$$j(q) = \begin{cases} 1/|B| & for \;\; q \in B \\ 0 & otherwise \end{cases}$$



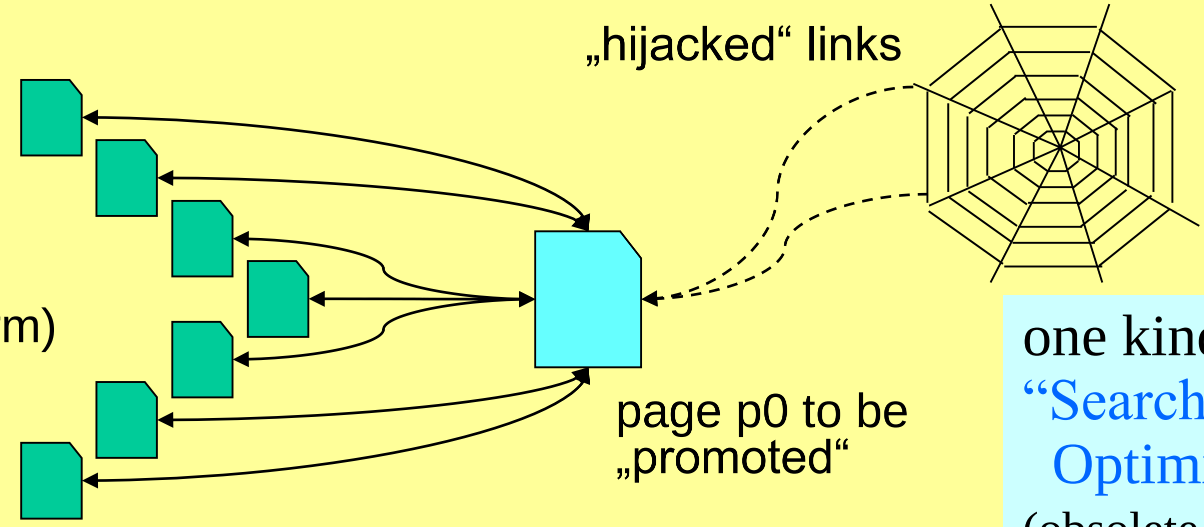
Authority (page q) = stationary prob. of visiting q

random walk: uniformly random choice of links + biased jumps to trusted pages

many other ways to **detect web spam** → **classifiers** etc.

Spam Farms and their Effect [Gyöngyi et al.: 2004]

Typical structure:



one kind of "Search Engine Optimization" (obsolete today)

Web transfers to p0 the „hijacked" score mass („leakage")
$\lambda = \Sigma_{q \in IN(p0)-\{p1..pk\}} PR(q)/outdegree(q)$

Theorem: p0 obtains the following PR authority:

$$PR(p0) = \frac{1}{1-(1-\varepsilon)^2}\left((1-\varepsilon)\lambda + \frac{\varepsilon((1-\varepsilon)k+1)}{n}\right)$$

The above spam farm is optimal within some family of spam farms (e.g. letting hijacked links point to boosting pages).

Countermeasures: TrustRank and BadRank

Gyöngyi et al.: 2004]

TrustRank:

start with explicit set T of trusted pages with trust values t_i

define random-jump vector r by setting $r_i = 1/|T|$ if $i \in T$ and 0 else

(or alternatively $r_i = t_i / \Sigma_{\nu \in T} t_\nu$)

propagate TrustRank mass to successors

$$TR(q) = \tau r + (1-\tau) \sum_{p \in IN(q)} TR(p) / outdegree(p)$$

BadRank:

start with explicit set B of blacklisted pages

define random-jump vector r by setting $r_i = 1/|B|$ if $i \in B$ and 0 else

propagate BadRank mass to predecessors

$$BR(p) = \beta r + (1-\beta) \sum_{q \in OUT(p)} BR(q) / indegree(q)$$

Problems:

maintenance of explicit lists is difficult

difficult to understand (& guarantee) effects

Link Analysis Without Links

[Kurland et al.: TOIS 2008]:
[Xue et al.: SIGIR 2003]

Apply simple data mining to **browsing sessions** of many users, where each session i is a sequence (pi_1, pi_2, ...) of **visited pages**:

- consider all pairs (pi_j, pi_{j+1}) of successively visited pages,
- compute their total frequency f, and
- select those with f above some min-support threshold

Construct **implicit-link graph** with the selected page pairs as edges and their normalized total frequencies f as edge weights
or construct graph from content-based **page-page similarities**

Apply **edge-weighted Page-Rank** for authority scoring, and linear combination of authority and content score etc.

Exploiting Click Log

[Chen et al.: WISE 2002]
[Liu et al.: SIGIR 2008]

Simple idea: Modify HITS or Page-Rank algorithm by **weighting edges** with the relative frequency of **users clicking on a link**

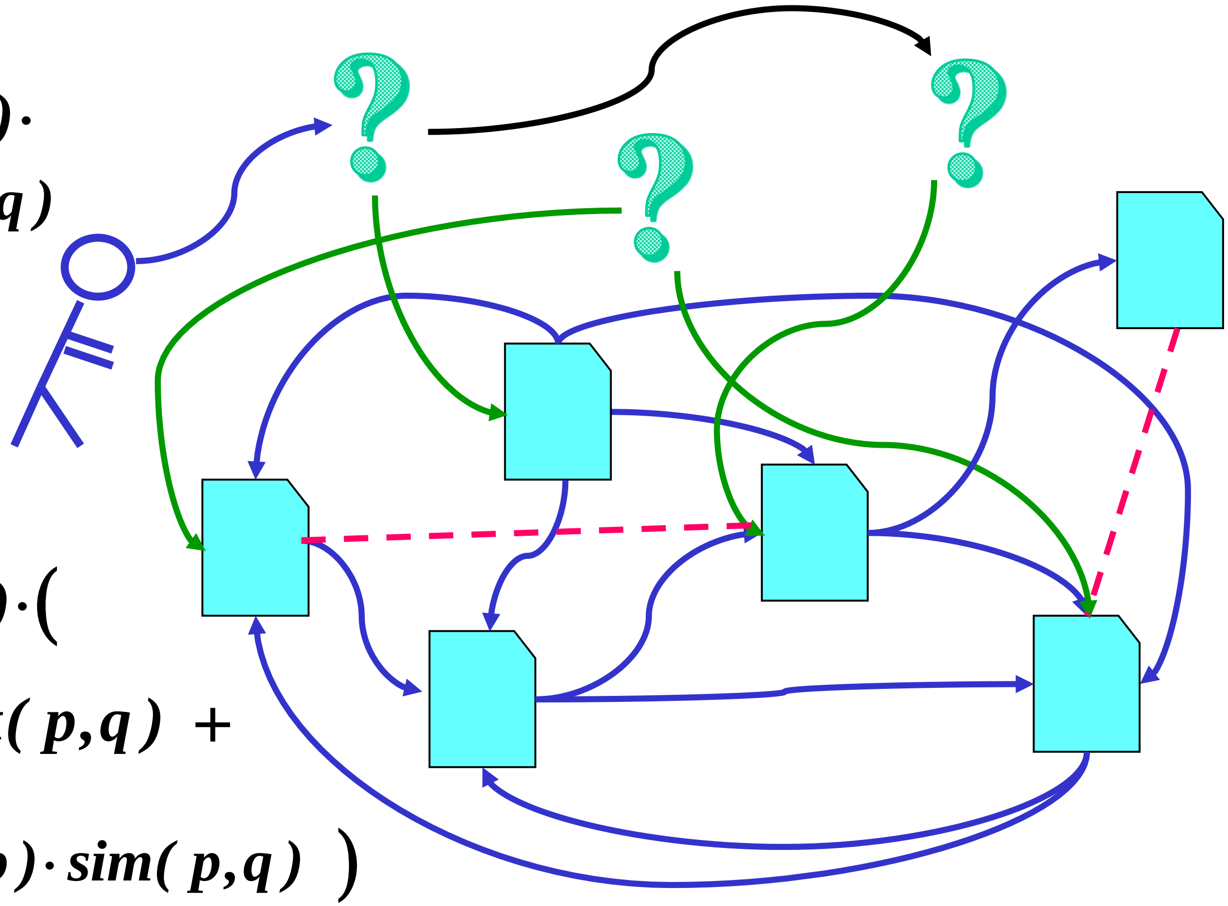
More sophisticated approach
Consider link graph A and
link-visit matrix V (V_{ij}=1 if user i visits page j, 0 else)
Define
authority score vector: $a = \beta A^T h + (1-\beta) V^T u$
hub score vector: $h = \beta A a + (1-\beta) V^T u$
user importance vector: $u = (1-\beta) V (a+h)$
with a tunable parameter β (β=1: HITS, β=0: DirectHit)

QRank: PageRank on Query-Click Graph

[Luxenburger et al.: WISE 2004]

<u>Idea:</u> add query-doc transitions + query-query transitions
+ doc-doc transitions on implicit links (by similarity)
with probabilities estimated from query-click log statistics

$$PR(q) = \varepsilon \cdot j(q) + (1-\varepsilon) \cdot \sum_{p \in IN(q)} PR(p) \cdot t(p,q)$$



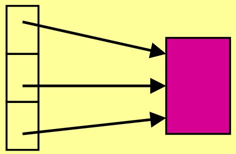
$$QR(q) = \varepsilon \cdot j(q) + (1-\varepsilon) \cdot \Big(\alpha \sum_{p \in explicitIN(q)} PR(p) \cdot t(p,q) + (1-\alpha) \sum_{p \in implicitIN(q)} PR(p) \cdot sim(p,q) \Big)$$

14.3 HITS: Hyperlink-Induced Topic Search

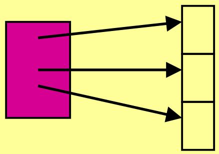
[J. Kleinberg: JACM 1999]

Idea:

Determine
- good content sources: **Authorities** (high indegree)



- good link sources: **Hubs** (high outdegree)



Find
- better authorities that have good hubs as predecessors
- better hubs that have good authorities as successors

For Web graph G = (V, E) define for nodes x, y $\in$ V

authority score $a_y \sim \sum_{(x,y) \in E} h_x$ and

hub score $h_x \sim \sum_{(x,y) \in E} a_y$

HITS as Eigenvector Computation

Authority and hub scores in matrix notation:

$$\vec{a} = \alpha E^T \vec{h} \qquad \vec{h} = \beta E \vec{a}$$

with constants α, β

Iteration with adjacency matrix A:

$$\vec{a} = \alpha E^T \vec{h} = \alpha\beta E^T E \vec{a} \qquad \vec{h} = \beta E \vec{a} = \alpha\beta E E^T \vec{h}$$

a and h are Eigenvectors of $E^T E$ and $E E^T$, respectively

<u>Intuitive interpretation:</u>

$M^{(auth)} = E^T E$ is the cocitation matrix: $M^{(auth)}_{ij}$ is the number of nodes that point to both i and j

$M^{(hub)} = E E^T$ is the bibliographic-coupling matrix: $M^{(hub)}_{ij}$ is the number of nodes to which both i and j point

HITS Algorithm

compute fixpoint solution by
iteration with length normalization:

initialization: $a^{(0)} = (1, 1, ..., 1)^T$, $h^{(0)} = (1, 1, ..., 1)^T$

repeat until sufficient convergence

$h^{(i+1)} := E\, a^{(i)}$

$h^{(i+1)} := h^{(i+1)} / \|h^{(i+1)}\|_1$

$a^{(i+1)} := E^T h^{(i)}$

$a^{(i+1)} := a^{(i+1)} / \|a^{(i+1)}\|_1$

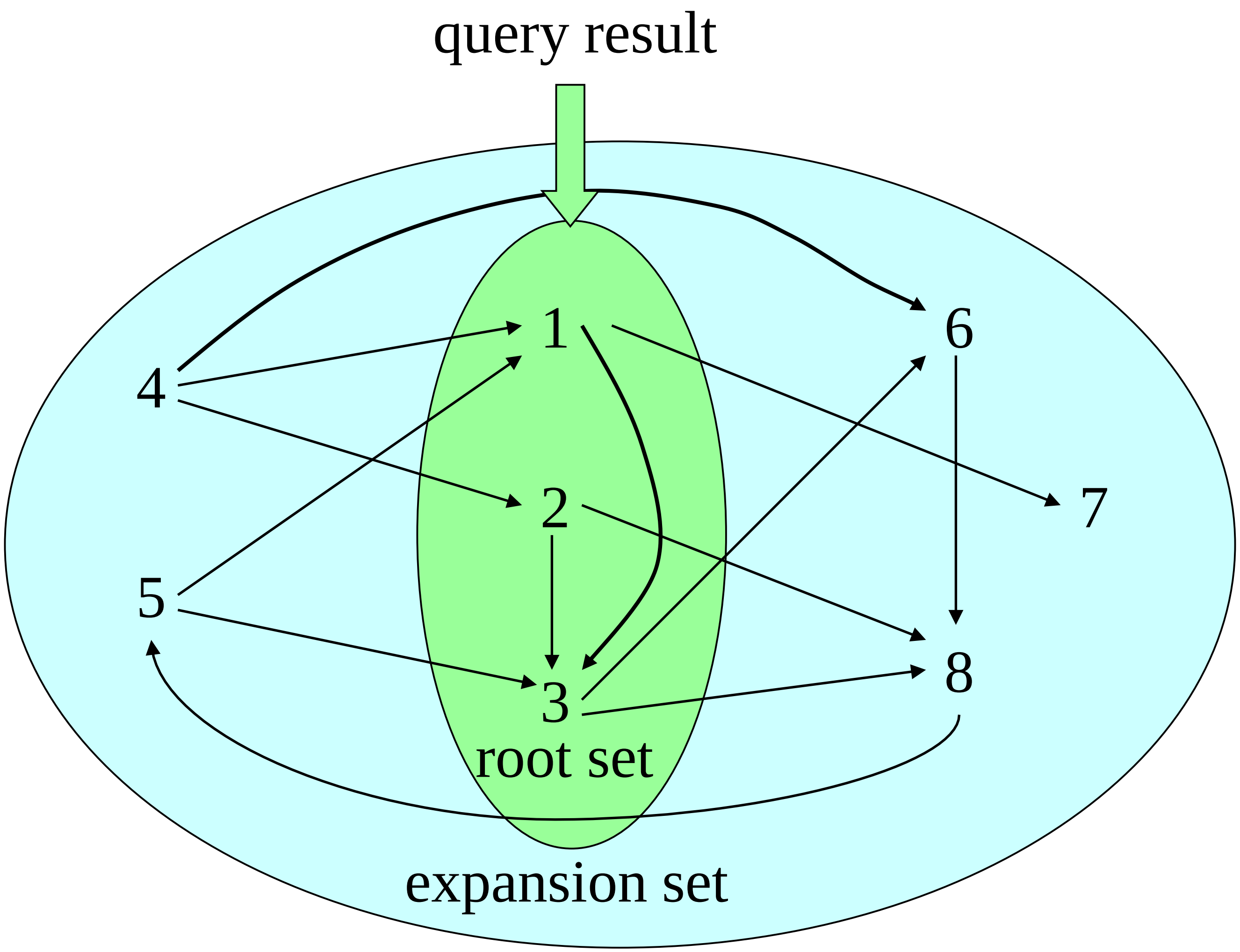
convergence guaranteed under fairly general conditions

Implementation of the HITS Algorithm

1) Determine sufficient number (e.g. 50-200) of „root pages“ via relevance ranking (e.g. tf*idf, LM …)
2) Add all successors of root pages
3) For each root page add up to d predecessors
4) Compute iteratively authority and hub scores of this „expansion set“ (e.g. 1000-5000 pages)
   with initialization $a_i := h_i := 1 / |\text{expansion set}|$
   and L_1 normalization after each iteration
   $\rightarrow$ converges to principal Eigenvector
5) Return pages in descending order of authority scores (e.g. the 10 largest elements of vector a)

„Drawback“ of HITS algorithm:
relevance ranking within root set is not considered

Example: HITS Construction of Graph



Enhanced HITS Method

Potential weakness of the HITS algorithm:

- irritating links (automatically generated links, spam, etc.)
- topic drift (e.g. from „python code“ to „programming“ in general)

Improvement:

- Introduce **edge weights**:
  0 for links within the same host,
  1/k with k links from k URLs of the same host to 1 URL *(aweight)*
  1/m with m links from 1 URL to m URLs on the same host (*hweight)*
- Consider **relevance weights** w.r.t. query topic (e.g. tf*idf, LM …)

→ Iterative computation of

authority score $$a_q := \sum_{(p,q)\in E} h_p \cdot \text{topicscore}(p) \cdot \text{aweight}(p,q)$$

hub score $$h_p := \sum_{(p,q)\in E} a_q \cdot \text{topicscore}(q) \cdot \text{hweight}(p,q)$$

Finding Related URLs

Cocitation algorithm:

- Determine up to B predecessors of given URL u
- For each predecessor p determine up to BF successors ≠ u
- Determine among all siblings s of u those with the largest number of predecessors that point to both s and u (degree of cocitation)

Companion algorithm:

- Determine appropriate base set for URL u („vicinity“ of u)
- Apply HITS algorithm to this base set

Companion Algorithm for Finding Related URLs

1) Determine **expansion set**: u plus
   - up to B predecessors of u and
     for each predecessor p up to BF successors ≠ u plus
   - up to F successors of u and
     for each successor c up to FB predecessors ≠ u

   with elimination of stop URLs (e.g. www.yahoo.com)

2) **Duplicate elimination:**
   Merge nodes both of which have more than 10 successors
   and have 95 % or more overlap among their successors

3) Compute **authority scores**
   using the improved HITS algorithm

HITS Algorithm for „Community Detection“

Root set may contain multiple topics or „communities“,
e.g. for queries „jaguar“, „Java“, or „randomized algorithm“

Approach:

- Compute k largest Eigenvalues of $E^T E$
  and the corresponding Eigenvectors a (authority scores)
  (e.g., using SVD on E)
- For each of these k Eigenvectors a
  the largest authority scores indicate
  a densely connected „community“

Community Detection more fully captured in Chapter 8

SALSA: Random Walk on Hubs and Authorities

[Lempel et al.: TOIS 2001]

View each node v of the link graph G(V,E) as two nodes v_h and v_a

Construct **bipartite undirected graph** G‘(V‘,E‘) from G(V,E):

$V' = \{v_h \mid v \in V \text{ and outdegree}(v)>0\} \cup \{v_a \mid v \in V \text{ and indegree}(v)>0\}$

$E' = \{(v_h, w_a) \mid (v,w) \in E\}$

Stochastic hub matrix H: $$h_{ij} = \sum_k \frac{1}{degree(i_h)} \frac{1}{degree(k_a)}$$

many other variants of link analysis methods

over all nodes with $(i_h,k_a), (k_a,j_h) \in E'$

Stochastic authority matrix A: $$a_{ij} = \sum_k \frac{1}{degree(i_a)} \frac{1}{degree(k_h)}$$

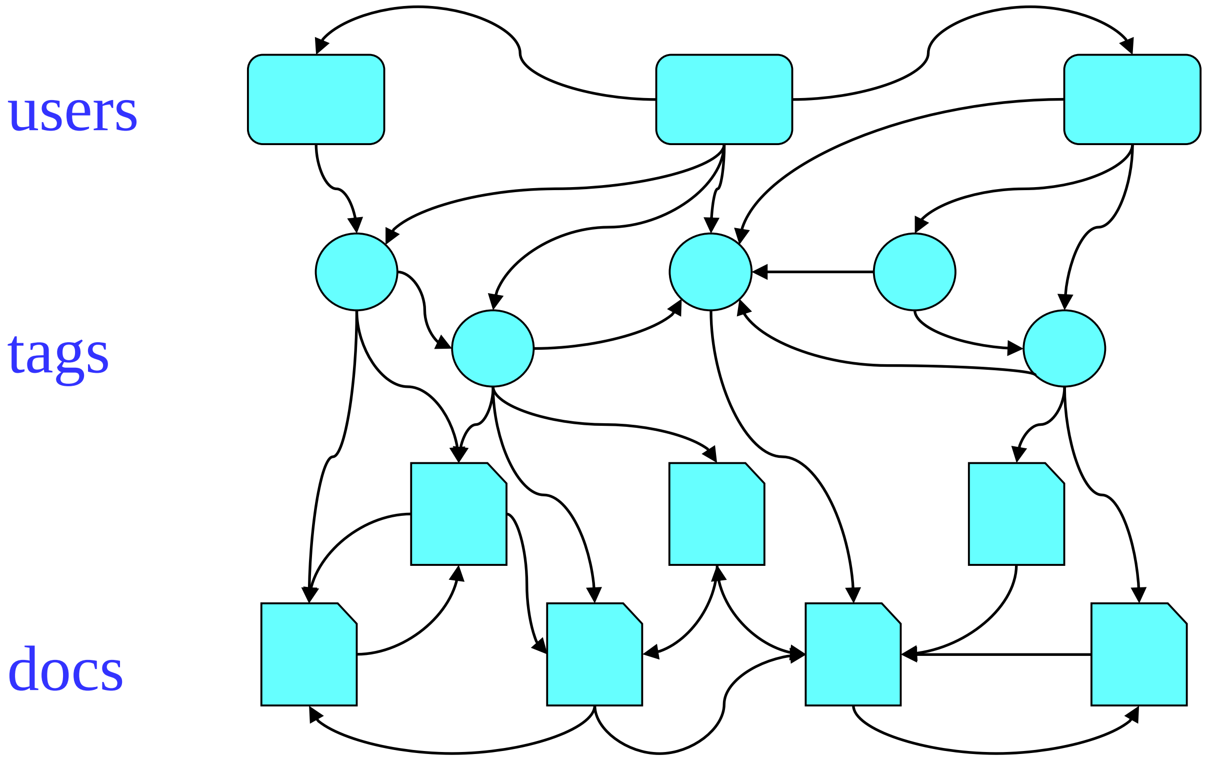
for i, j and k ranging over all nodes with $(i_a,k_h), (k_h,j_a) \in E'$

The corresponding Markov chains are ergodic on connected component

Stationary solution: $\pi[v_h] \sim$ outdegree(v) for H, $\pi[v_a] \sim$ indegree(v) for A

Further extension with random jumps: **PHITS (Probabilistic HITS)**

14.4 Extensions for Social & Behavioral Graphs



Typed graphs: data items, users, friends, groups, postings, ratings, queries, clicks, …
with weighted edges

Social Tagging Graph

Tagging relation in **„folksonomies“**:

- ternary relationship between users, tags, docs
- could be represented as hypergraph or tensor
- or (lossfully) decomposed into 3 binary projections (graphs):

UsersTags (<u>UId, TId</u>, UTscore)

x.UTscore := Σ_d {s | (x.UId, x.TId, d, s) $\in$ Ratings}

TagsDocs (<u>TId, Did</u>, TDscore)

x.TDscore := Σ_u {s | (u, x.TId, x.DId, s) $\in$ Ratings}

DocsUsers (<u>DId, UId</u>, DUscore)

x.DUscore := Σ_t {s | (x.UId, t, x.DId, s) $\in$ Ratings}

Authority/Prestige in Social Networks

Apply link analysis (PR, PPR, HITS etc.) to appropriately defined matrices

- **SocialPageRank** [Bao et al.: WWW 2007]:

  Let M_{UT}, M_{TD}, M_{DU} be the matrices corresponding to relations UsersTags, TagsDocs, DocsUsers

  Compute iteratively with renormalization:

$$\vec{r}_T = M_{UT}^T \times \vec{r}_U$$
$$\vec{r}_D = M_{TD}^T \times \vec{r}_T$$
$$\vec{r}_U = M_{DU}^T \times \vec{r}_D$$

- **FolkRank** [Hotho et al.: ESWC 2006]:

  Define ***graph G as union of graphs*** UsersTags, TagsDocs, DocsUsers

  Assume each user has personal preference vector $\vec{p}$

  Compute iteratively: $\vec{r}_D = \alpha \vec{r}_D + \beta M_G \times \vec{r}_D + \gamma \vec{p}$

Search & Ranking with Social Relations

Web search (or search in social network incl. enterprise intranets) can benefit from the taste, expertise, experience, recommendations of friends and colleagues

$\rightarrow$ use social neighborhood for query expansion, etc.

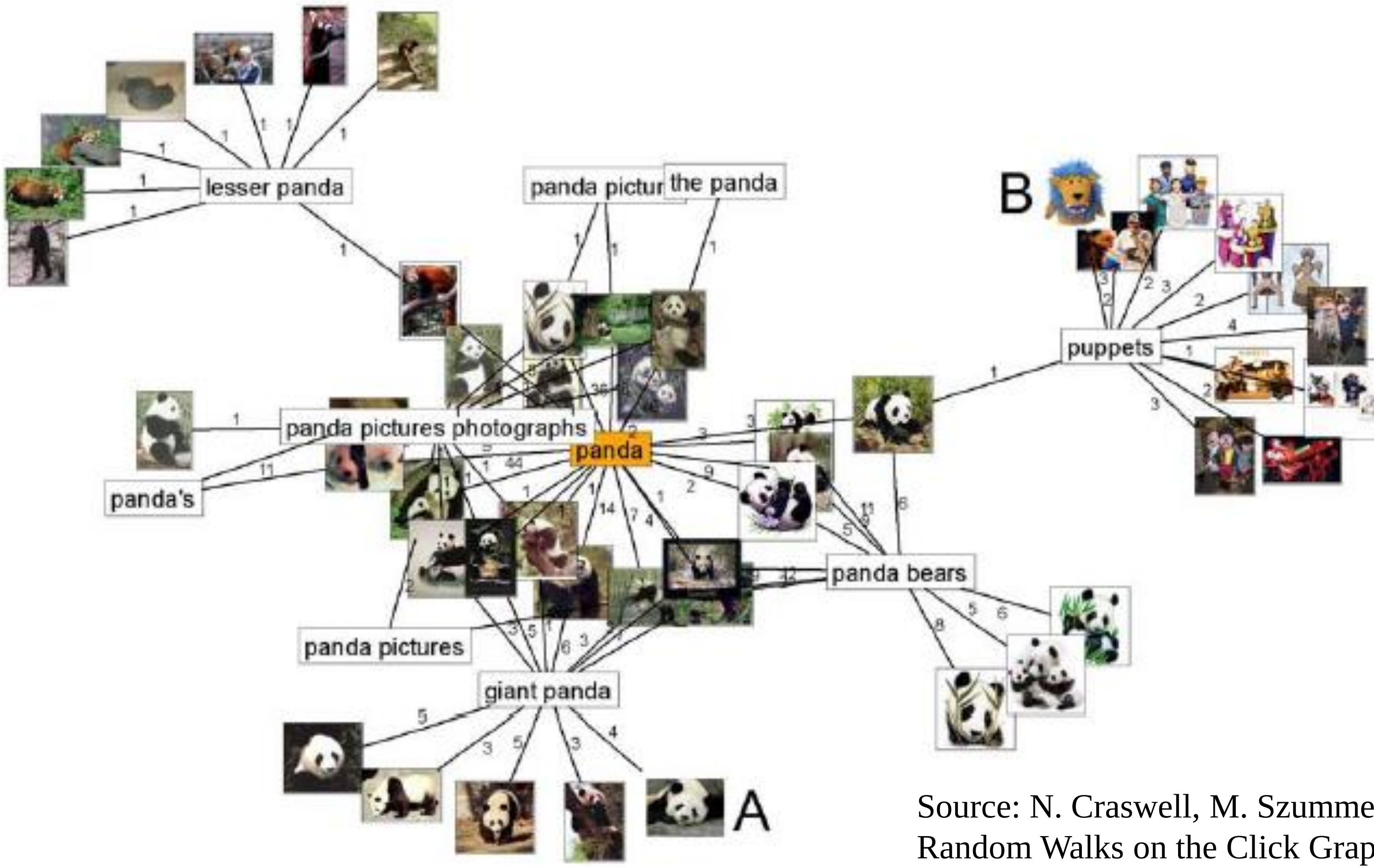
$\rightarrow$ combine content scoring with FolkRank, SocialPR, etc.

$\rightarrow$ integrate friendship strengths, tag similarities, community behavior, individual user behavior, etc.

$\rightarrow$ further models based on random walks for twitter followers, review forums, online communities, etc.

Random Walks on Query-Click Graphs

Bipartite graph with queries and docs as nodes and edges based on clicks with weights ~ click frequency



Source: N. Craswell, M. Szummer: Random Walks on the Click Graph, SIGIR 2007

Random Walks on Query-Click Graphs

Bipartite graph with queries and docs as nodes and edges based on clicks with weights ~ click frequency

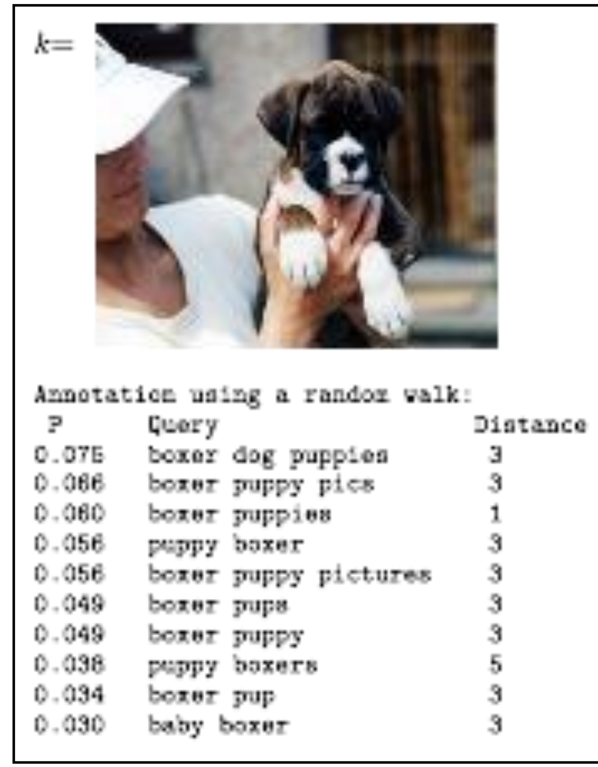
[Craswell: SIGIR'07]

transition probabilities:

$t(q,d) = (1\text{-}s)\ C_{qd} / \Sigma_i C q_i$ for $q \neq d$
with click frequencies C_{qd}
$t(q,q) = s$ with self-transitions

Useful for:

- query-to-doc ranking
- query-to-query suggestions
- doc-to-query annotations
- doc-to-doc suggestions



Example: doc-to-query annotations

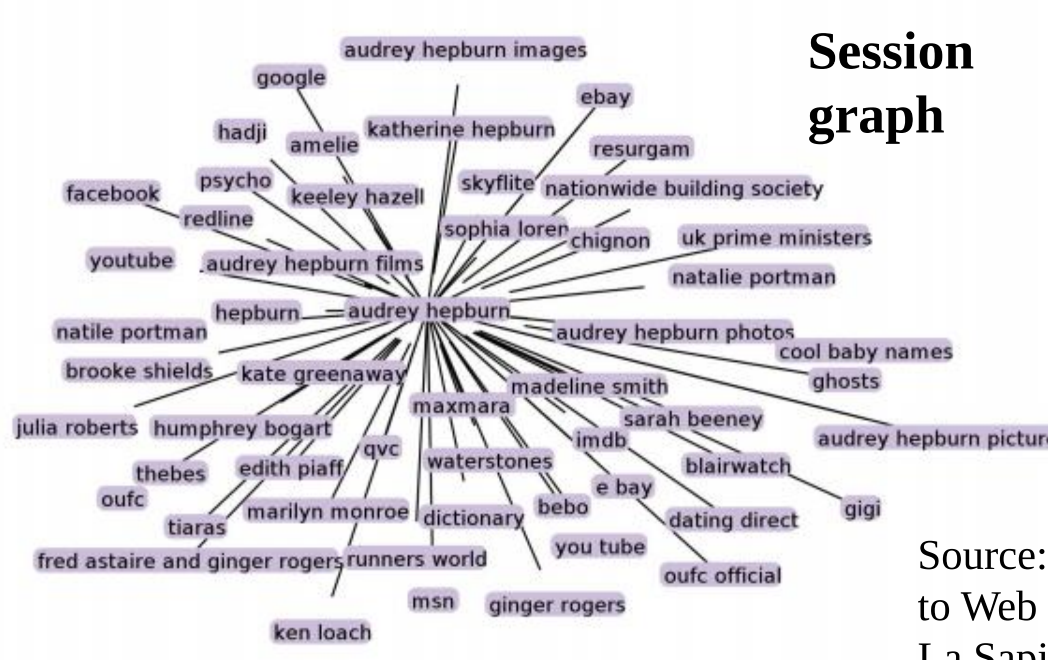
Query Flow Graphs

[Boldi et al.: CIKM'08, Bordino et al.: SIGIR'10]

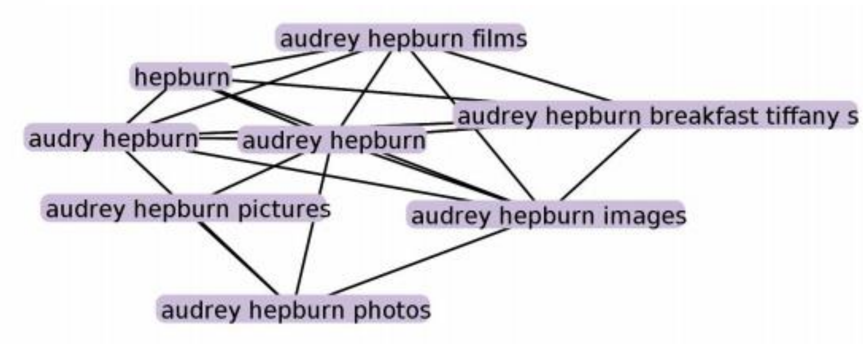
Graph with queries as nodes and edges derived from user sessions (query reformulations, follow-up queries, etc.)

transition probabilities: t(q,q') ~ P[q and q' appear in same session]

Session graph



Click graph



Source: Ilaria Bordino, Graph Mining and its applications to Web Search, Doctoral Dissertation, La Sapienza University Rome, 2010

Link analysis yields suggestions for query auto-completion, reformulation, refinement, etc.

Summary of Chapter 14

- **PageRank** (PR), **HITS**, etc. are elegant models for
  query-independent page/site authority/prestige/importance
- Query result ranking combines PR with content
- Many **interesting extensions** for
  personalization (RWR), query-click graphs, doc-doc similarity etc.
- Potentially interesting for ranking/recommendation in **social networks**
- **Random walks** are a powerful instrument

Additional Literature for 14.1 and 14.3

- S Brin, L.Page: Anatomy of a Large-Scale Hypertextual Web Search Engine, WWW 1998
- L. Page, S. Brin, R. Motwani, L. Page, T. Winograd: The PageRank Citation Ranking: Bringing Order to the Web, Technical Report, Stanford University, 1997
- M. Bianchini, M. Gori, F. Scarselli: Inside PageRank, TOIT 5(1), 2005
- A.N. Langville, C.D. Meyer: Deeper inside PageRank. Internet Math., 1(3), 2004
- A. Broder et al.: Efficient PageRage Approximation via Graph Aggregation. Inf. Retr. 2006
- G. Jeh, J. Widom: SimRank: a Measure of Structural-Context Similarity, KDD 2002
- D. Fogaras, B. Racz:: Scaling link-based similarity search. WWW 2005
- J.M. Kleinberg: Authoritative Sources in a Hyperlinked Environment, JACM 1999
- K. Bharat, M. Henzinger: Improved Algorithms for Topic Distillation in a Hyperlinked Environment, SIGIR 1998
- R.Lempel et al.: SALSA: Stochastic Approach for Link-Structure Analysis, TOIS 19(2), 2001
- J. Dean, M. Henzinger: Finding Related Pages in the WorldWideWeb, WWW 1999
- A. Borodin et al.: Link analysis ranking: algorithms, theory, and experiments. TOIT 5(1), 2005
- M. Najork et al.: :Hits on the web: how does it compare? SIGIR 2007

Additional Literature for 14.2 and 14.4

- Taher Haveliwala: Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search, IEEE Trans. on Knowledge and Data Engineering, 2003
- G. Jeh, J. Widom: Scaling personalized web search, WWW 2003.
- Z. Gyöngyi, H. Garcia-Molina: Combating Web Spam with TrustRank, VLDB‘04.
- Z. Gyöngyi et al.: Link Spam Detection based on Mass Estimation, VLDB‘06
- Z. Chen et al.: A Unified Framework for Web Link Analysis, WISE 2002
- Y. Liu et al.: BrowseRank: letting web users vote for page importance. SIGIR 2008
- G.-R. Xue et al.:: Implicit link analysis for small web search,. SIGIR 2003
- O. Kurland, L. Lee: PageRank without hyperlinks: Structural reranking using links induced by language models. ACM TOIS. 28(4), 2010
- S. Bao et al.: Optimizing web search using social annotations, WWW 2007
- A. Hotho et al.: Information Retrieval in Folksonomies: Search and Ranking. ESWC 2006
- J. Weng et al.: TwitterRank: finding topic-sensitive influential twitterers, WSDM 2010
- N. Craswell, M. Szummer: Random walks on the click graph, SIGIR 2007
- P. Boldi et al.: The query-flow graph: model and applications, CIKM 2008
- I. Bordino et al.: Query similarity by projecting the query-flow graph, SIGIR 2010