Google's eigenvector

The secret of PageRank

Adhemar Bultheel

Dept. Computer Science, K.U.Leuven

10th October 2007

Survey

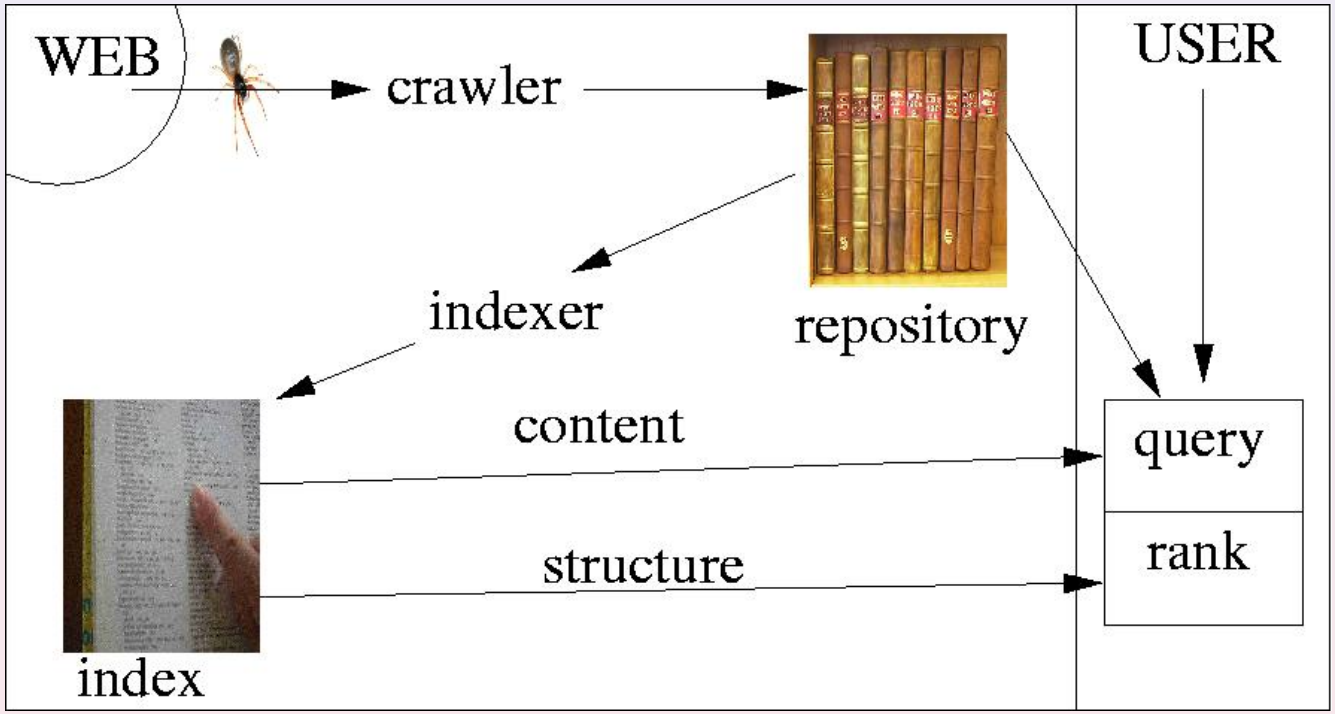
- The players
- Link analysis
- PageRank = Google's eigenvector
- Properties and computation

Properties

The web is

- Huge ($10^{10} - 10^{11}$ pages on surface; many more in the deep web)
- Dynamic (40% changes within a week)
- Self organized (no central administration)
- Hyperlinked (linkanalysis can be used to find a relevant item)

The mechanism



- Crawler (send out spider robots)
- Indexing (inverted file)
  (in 2004 Google needed 15000 computers to store it)
- Too many results, hence rank the results.

1998 takeoff





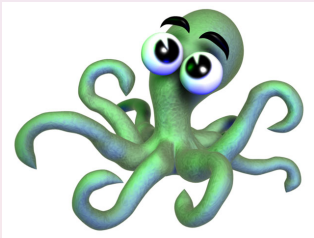
HITS (Hypertext Induced Topic Search)
by Jon Kleinberg at IBM Silicon Valley
Now professor at Cornell
presented at ACM-SIAM meeting on
Discrete algorithms (San Diego)

PageRank
by Larry Page and Sergey Brin at Stanford U.
Bachelor students since 1995, start up Google
presented at WWW meeting in Australia

HITS Thesis

Importance is earned from others

hub
many ougoing links
(outlinks)



authority
many incoming links
(inlinks)



ranking:
a good hub points to good authorities
a good authority is pointed to by good hubs
Developed in 1997-98; implemented in Teoma 2001 (now Ask)

PageRank Thesis

A page is important if many important pages refer to it

Importance is defined by the self-regulating system of the web.
The web is democratic.
Your inlinks define your importance.
This importance you can distribute over your outlinks.
your inlinks

PageRank Thesis



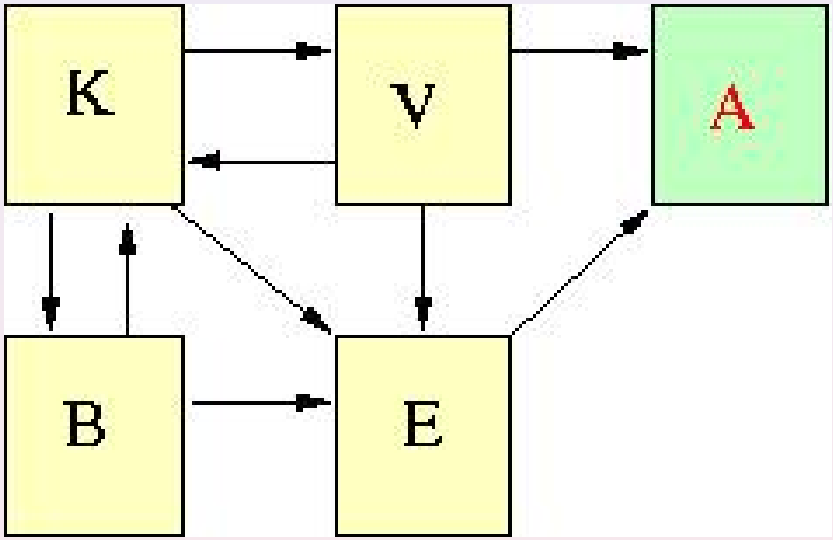
The democracy of the web, pages vote for pages.

ranking:

This is an eigenvalue problem.

Can be solved by random walk (Markov chain).

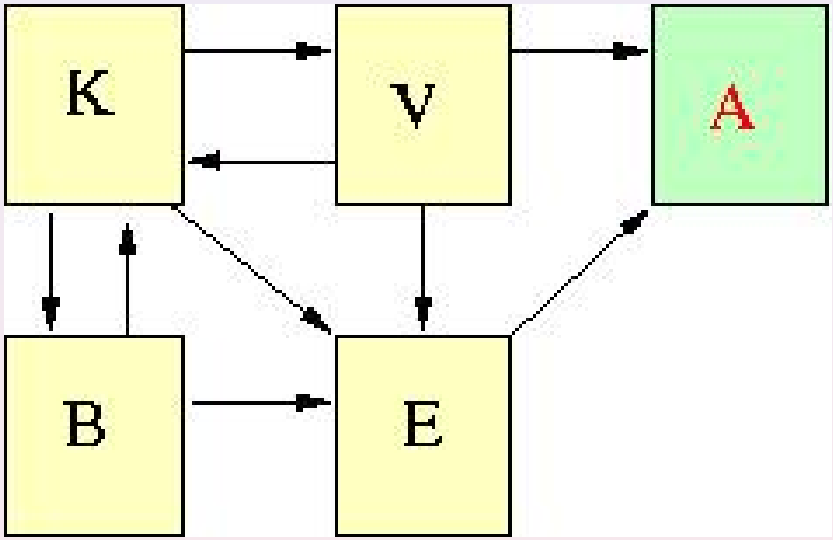
A toy example



	K	V	A	B	E
K	0	1/3	0	1/3	1/3
V	1/3	0	1/3	0	1/3
A	0	0	0	0	0
B	1/2	0	0	0	1/2
E	0	0	1	0	0

1. Every page gets franchise value of 1 vote
2. Equally distribute its franchise value over outlinks
3. After vote: new franchise value to be distributed
4. Continue with step 2
5. until convergence (?)

A toy example

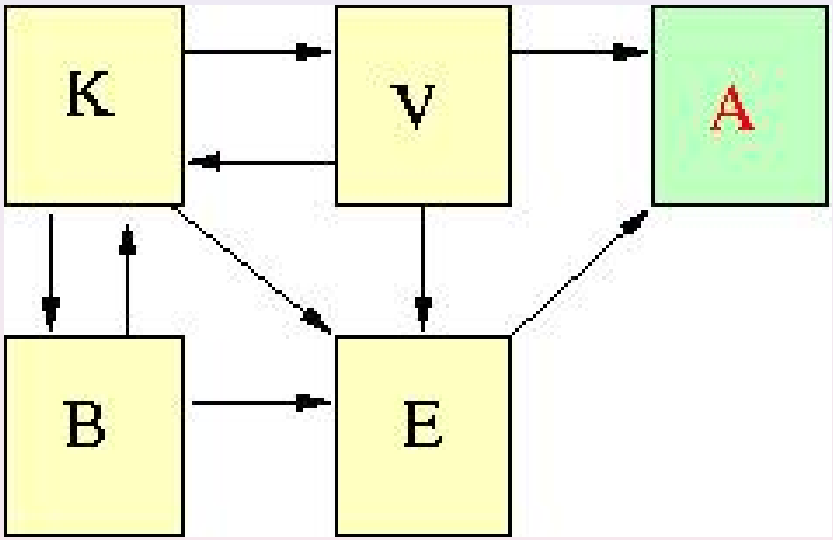


	K	V	A	B	E
K	0	1/3	0	1/3	1/3
V	1/3	0	1/3	0	1/3
A	0	0	0	0	0
B	1/2	0	0	0	1/2
E	0	0	1	0	0

$= H$

- Sum of rows = what is distributed by row page
- Sum of columns is what is received by column page
- π_k is the state of the values after vote k
- E.g. $\pi_0 = [1\ 1\ 1\ 1\ 1]$
- $\pi_{k+1} = \pi_k H$; e.g. $\pi_1 = [5/6\ 1/3\ 4/3\ 1/3\ 7/6]$

A toy example

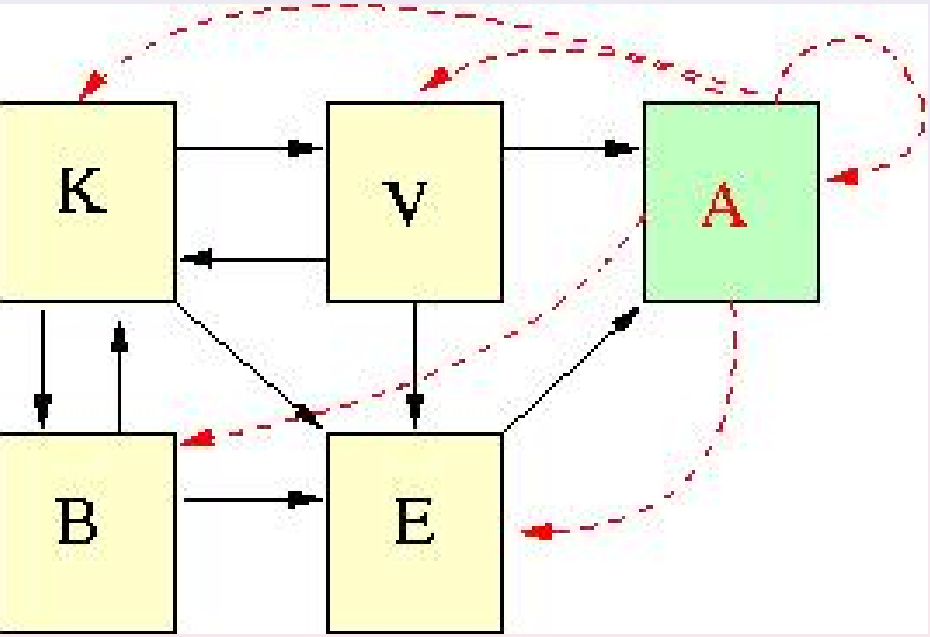


	K	V	A	B	E
K	0	1/3	0	1/3	1/3
V	1/3	0	1/3	0	1/3
A	0	0	0	0	0
B	1/2	0	0	0	1/2
E	0	0	1	0	0

$= H$

- $\pi_{k+1} = \pi_k H$
- note A is a *dangling* page (no outlinks)
- hence a zero row in *hyperlink matrix* H
- other rows have sum 1 (= probability distribution)
- H is huge and sparse
- does π_k converge to π (= *PageRank vector*)?

Problems ⇒ random walk



	K	V	A	B	E
K	0	1/3	0	1/3	1/3
V	1/3	0	1/3	0	1/3
A	1/5	1/5	1/5	1/5	1/5
B	1/2	0	0	0	1/2
E	0	0	1	0	0

$= S$

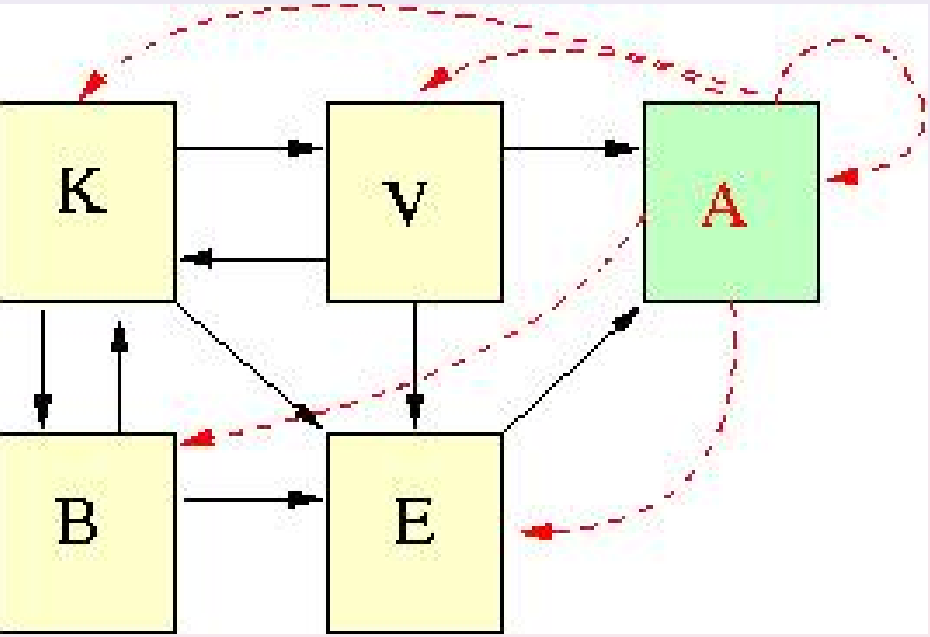
- Dangling pages are a problem (black hole for votes)
- A surfer stuck on a dangling page could be *teleported* to any page at random according to some probability distribution.
- now *all* the rows of S sum to 1.
- with probability α teleport or outlink on any page

Markov chains

$$\boxed{\pi_{k+1} = \pi_k G,\ G = \alpha S + (1-\alpha)E}$$

- G is the *Google matrix*
- E is *teleport matrix*, e.g. $E = (1/n)e^T e$, $e = [1, 1, \ldots, 1]$
- $S = H + (1/n)a^T e$ (a binary vector to mark dangling pages)
- Google takes $\alpha = 0.85$
- G is row-stochastic matrix ($G_{ij} \geq 0$, $\sum_j G_{ij} = 1$)
- the process converges to a unique PageRank vector π which gives a probability distribution
- π is dominant eigenvector (largest eigenvalue $= 1$)
- PR on log(?) scale from 0 to 10 (= google page)

Results for example



	K	V	A	B	E
K	0	1/3	0	1/3	1/3
V	1/3	0	1/3	0	1/3
A	1/5	1/5	1/5	1/5	1/5
B	1/2	0	0	0	1/2
E	0	0	1	0	0

$= S$

k	K	V	A	B	E
1	0.2056	0.1206	0.2906	0.1206	0.2623
2	0.1648	0.1376	0.3365	0.1376	0.2231
3	0.1847	0.1339	0.3159	0.1339	0.2314
4	0.1785	0.1360	0.3183	0.1360	0.2309
5	0.1804	0.1347	0.3189	0.1347	0.2310
6	0.1796	0.1350	0.3188	0.1353	0.2307
7	0.1800	0.1351	0.3187	0.1351	0.2309
8	0.1798	0.1352	0.3187	0.1352	0.2309
9	0.1799	0.1351	0.3187	0.1351	0.2309
10	0.1799	0.1358	0.3187	0.1351	0.2309

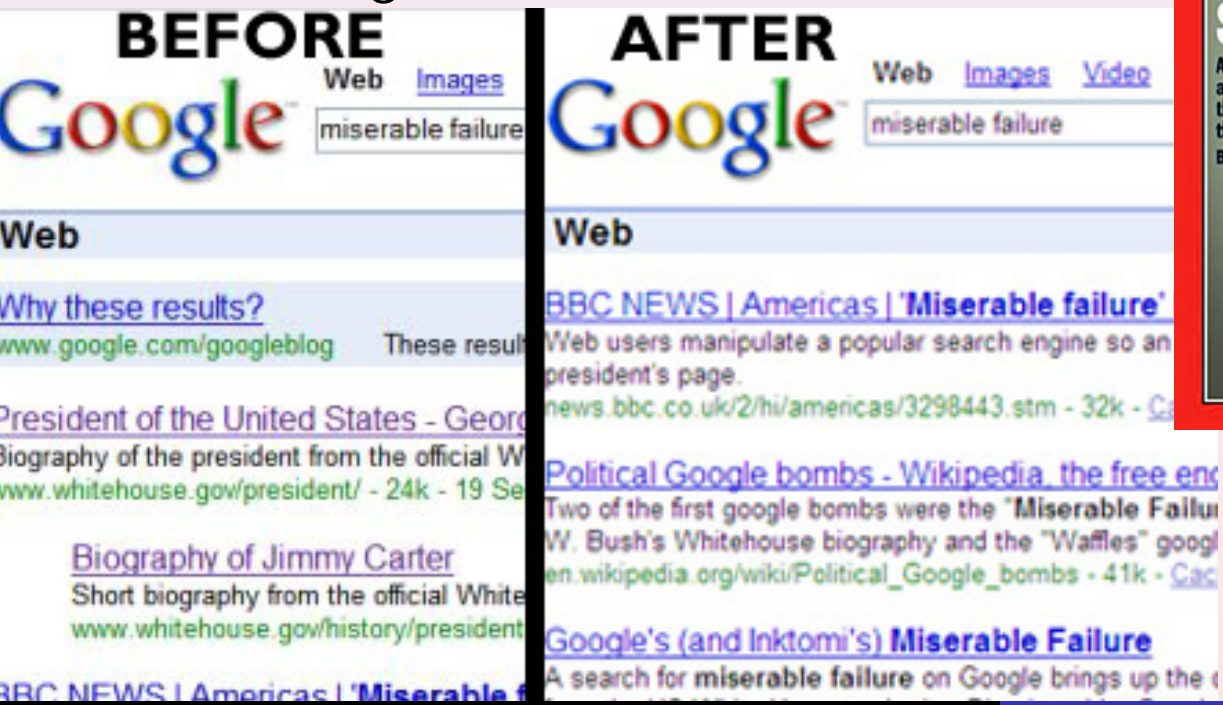
(see maple)

$\alpha = 0.85,\ E = 1/5 e^T e.$

Google's PageRank (the $25,000,000,000 eigenvector)

- Page refers to Larry Page (?)
- Success of Google (public 2004)
- SEO (Search Engine Optimizers) industry (SearchKing)
- link farms to increase PR
- Google bombs





Some examples

- my home
- department
- faculty
- kvab
- kuleuven
- ugent
- google

Personalized teleport

- personalized page ranking ultimate goal of companies but...
- Kaltix technology bought by Google in 2003
  (Glen Jeh, Sepandar Kamvar, Taher Haveliwala) @ Stanford U.
- $G = \alpha S + (1-\alpha)E = \alpha H + [\alpha a^T + (1-\alpha)e^T]v$
- $\Rightarrow \pi_{k+1} = \pi_k G = \alpha \pi_k H + [\alpha \pi_k a^T + (1-\alpha)]v$
- !! it takes days to compute $\pi(v)$ (a PR for particular v)
- topic sensitive: $\pi = \sum \beta_i \pi(v_i)$ $i \in \{\text{sports, news, arts,...}\}$
- Personalized google search (teleport vector v) iGoogle
- query sensitive (see amazon)





Problems

- How to store G when of order 10^{11}?
- How accurate should π be?
- How often to update the PR?
- Can we speed up the process?
- Which method to use?
- How sensitive is PR for the parameters?

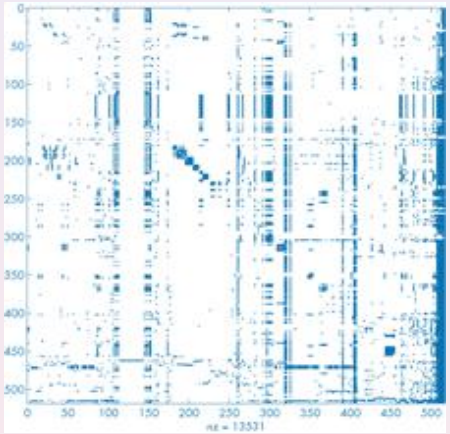
Huge scale problem

The world largest matrix computation (Cleve Moler)

- n = number of web pages ($8.1 \cdot 10^9$)
- H = hyperlink matrix (n^2 elements but sparse)
- $\varnothing(H)$ = # nonzeros in H
  (# outlinks per page is about 10 $\Rightarrow \varnothing(H) \approx 10n$)
- d = # dangling nodes
- a = has d entries =1
- v = personalization teleport vector (n)
- π = PageRank vector (n)
- $\pi_{k+1} = \alpha\pi_k H + (\alpha\pi_k a^T + 1 - \alpha)v$
- $\pi_k H$ requires $\varnothing(H)$ flops

Huge scale problem

The world largest matrix computation (Cleve Moler)



sparse matrix 512×512

For a matlab implementation see the surfer.m script at Moler's site.

Rows stored as *adjacency lists* with *data compression*

in same domain often similar outlinks
large gaps in link lists
$\Rightarrow$ compress

Haveliswala proposes to compress π so that it can stay in cache, hence fast reaction time.

Precision and convergence

- Accurate π is not important, it suffices to obtain the right order
- After 10 iterations the ordering is already correct (P&B report 50 iterations)
- Can one iterate with "orderings" instead of with the real π?
- Google computes "finer" rankings than the PR0:PR10
- speed of convergence of power method depends on gap $\lambda_1/\lambda_2 = 1/\lambda_2$. Depends on α, which should not be close to 1!

$$\|\pi_k - \pi\|_1 \leq \alpha^k \|\pi_0 - \pi\|_1$$

- stopping criterion $\|\pi_k - \pi\| \leq n\tau$ (its $\downarrow$ 50%, disagree $\leq$ 1.5%)

Sensitivity is not an issue ...

... if α is not too close to 1.

- $\|\Delta\pi\|_1 \leq \frac{\alpha}{1-\alpha}\|\Delta S\|_\infty$
- $\|\Delta\pi\|_1 \leq \frac{2}{1-\alpha}\Delta\alpha$
- $\|\Delta\pi\|_1 \leq \|\Delta v\|_1$

PageRank as a linear system

- $\pi \underbrace{(\alpha S + (1-\alpha) e^T v)}_{G} = \pi, \quad \pi \geq 0, \quad \|\pi\|_1 = \pi e^T = 1$
- $\Rightarrow \pi(I - \alpha S) = (1-\alpha)v, \quad$ note $(I - \alpha S)$ nonsingular
- huge system to be solved iteratively (e.g. Jacobi)
- convergence can be faster, and holds even for $\alpha = 1$.
- $S = H + aw^T$ is dense (!) since zero rows of dangling nodes are filled up (we took $w = v$ before)
  and thus have to invert the dense matrix $I - \alpha S$
  - w defines escape from dangling page
  - v defines random jump from any page.

Dangling nodes

- dangling page has no outlinks
- pages "yet-to-be-crawled" are also dangling.
- Dangling nodes = multiply by 0.
  - P&B propose to remove them and insert them back in only for final iterations. They can be very important though. See toy example.
  - Lee et al propose to collapse all dangling states into one. Then no ranking for dangling states, and biased ranking for others.
  - Langville & Meyer remove the $a^T w$. Then $H \neq S$ is not stochastic (has zero rows). Numerical problems!

Dangling nodes

- If $v = w$ then system is

$$\pi(I - \alpha S) = (1-\alpha)v \Rightarrow \pi \underbrace{(I - \alpha H)}_{\text{sparse}} = \underbrace{(1 - \alpha + \alpha \pi a^T)v}_{\text{multiple of } v}$$

  Hence solve $\tilde{\pi}(I - \alpha H) = v$ and $\tilde{\pi}$ is a multiple of π
- If $v \neq w$ then $I - \alpha S = (I - \alpha H) - \alpha a^T w$.
  - solve $\delta(I - \alpha H) = (1-\alpha)v$
  - update $\pi = \delta + \Delta$

Dangling nodes

- solve $\delta(I - \alpha H) = (1 - \alpha)v$

  permutation to separate dangling from nondangling

  $$\underbrace{[\delta_1 \ \delta_2]}_{\delta} \begin{bmatrix} I - \alpha H_1 & I - \alpha H_2 \\ 0 & I \end{bmatrix} = (1 - \alpha) \underbrace{[v_1 \ v_2]}_{v}$$

  - sparse solve $\delta_1(I - \alpha H_1) = (1 - \alpha)v_1$
  - $\delta_2 = \alpha \delta_1 H_2 + (1 - \alpha)v_2$

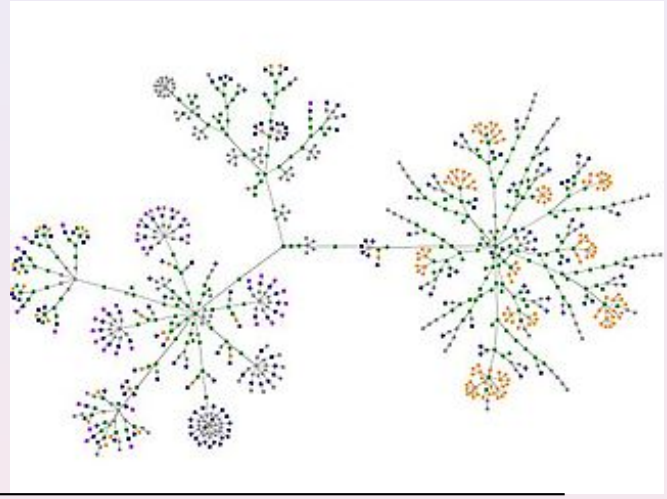
- update $\pi = \delta + \Delta$

  write $a^T w = \begin{bmatrix} 0 \\ e \end{bmatrix} [w_1 \ w_2]$

  - sparse solve $z(I - \alpha H_1) = \alpha w_1$
  - $y = \alpha(w_2 + zH_2)$
  - $\pi = \delta + \underbrace{\dfrac{\|\delta_2\|_1}{1 - \|y\|_1}[z \ y]}_{\Delta}$

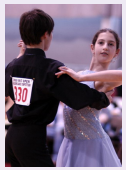
BlockRank by aggregation

- To reduce the work per iteration and accelerate the convergence
- Intranets have many intranet links and only few interlinks



- compute PR Π for portal pages of intranets (ingnore intranet links)
- compute PR $\pi^{[i]}$ per intranet (ignoring interlinks)
- BR of page in intranet i is $\pi^{[i]} \cdot \Pi(i)$
- this is an approximation of PR π because certain links are ignored
- gives a speed-up of 2

Google dance





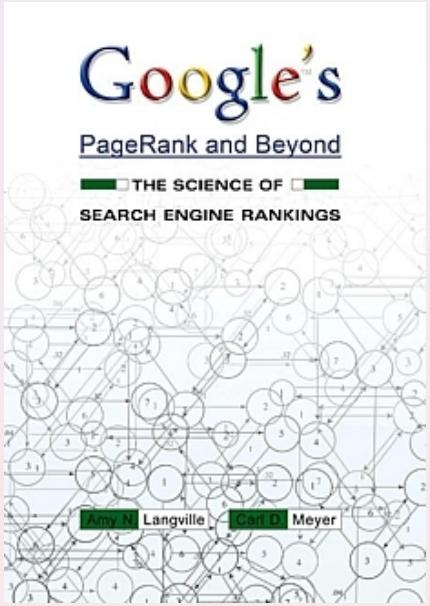
monthly update of π

- hyperlinks may chance (content of G changes = link update)
- pages may be added or deleted (size of G changes = page update)
- uses aggregation (block updates) (start from old π is not an option)
- index distributed over 10,000 servers (linux PCs)
- google dance tool shows links in 3 google servers: www—2—3.google.com: main one and 2 test servers
- GDS (google dance syndrom) in 2003 major changes by google
- yearly google dance party

Ongoing research

- Other methods are investigated (Krylov subspace method, Gauss-Seidel,...)
- Parallel algorithms
- Modelling the "back" button of the browser
- Query or time dependent ranking (?)
- Other ranking procedures (PageRank+HITS=SALSA, traffic flow,...)
- One step behind (link)spammers
- ...

Reference



A.N. Langville en C.D. Meyer
Google's PageRank and Beyond, the science of search engine rankings
Princeton University Press (July 3, 2006)