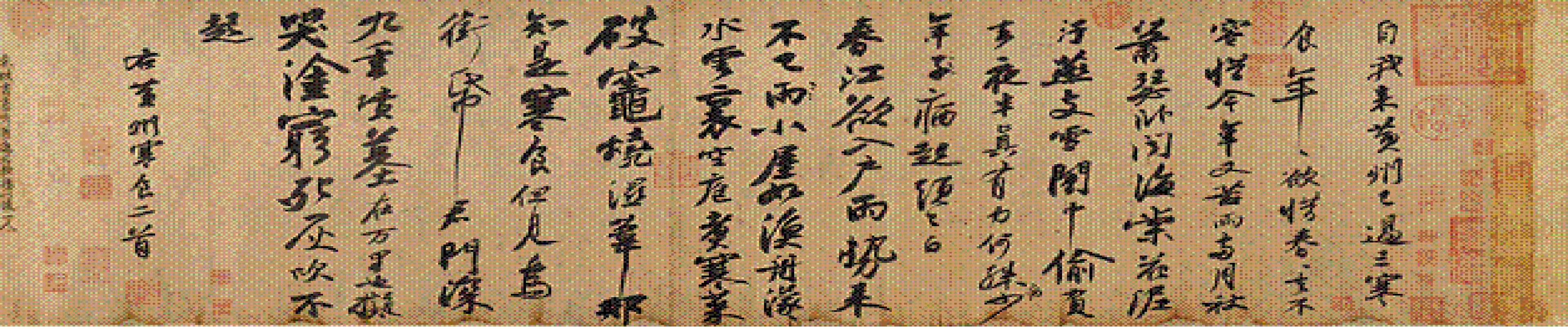


信息检索
Information Retrieval

教师：孙茂松

Tel:62781286

Email:sms@tsinghua.edu.cn

TA：**胡锦毅**

Email:hu-jy21@mails.tsinghua.edu.cn

郑重声明

- 此课件仅供选修清华大学计算机系本科生课《信息检索》（课号：40240372）的学生个人学习使用，所以只允许学生将其下载、存贮在自己的电脑中。未经孙茂松本人同意，任何人不得以任何方式扩散之（包括放到9#服务器上）。否则，由此可能引起的一切涉及知识产权的法律责任，概由该人负责。
- 此课件仅限孙茂松本人讲课使用。除孙茂松本人外，凡授课过程中，PTT文件显示此《郑重声明》之情形，即为侵权使用。

第七章 Web信息检索

The World Wide Web

- Developed by Tim Berners-Lee in 1990 at CERN to organize research documents available on the Internet.
- Combined idea of documents available by FTP with the idea of *hypertext* to link documents.
- Developed initial HTTP network protocol, URLs, HTML, and first "web server."

Web Pre-History

Ted Nelson developed idea of hypertext in 1965.

Doug Engelbart built the first implementation of hypertext in the late 1960's at SRI.

The basic technology was in place in the 1970's; but it took the PC revolution and widespread networking to inspire the web and make it practical.

Web Browser History

- Early browsers were developed in 1992 (Erwise, ViolaWWW).
- In 1993, Marc Andreessen and Eric Bina at UIUC NCSA developed the Mosaic browser and distributed it widely.
- Andreessen joined with James Clark (Stanford Prof. and Silicon Graphics founder) to form Mosaic Communications Inc. in 1994 (which became Netscape to avoid conflict with UIUC).
- Microsoft licensed the original Mosaic from UIUC and used it to build Internet Explorer in 1995.

Web Search History

- In 1993, early web robots (spiders) were built to collect URL's:
  - Wanderer
  - ALIWEB (Archie-Like Index of the WEB)
  - WWW Worm (indexed URL's and titles for regex search)
- In 1994, Stanford grad students David Filo and Jerry Yang started manually collecting popular web sites into a topical hierarchy called Yahoo.

Web Search History

- In early 1994, Brian Pinkerton developed WebCrawler as a class project at U Wash. (eventually became part of Excite and AOL).
- A few months later, Fuzzy Maudlin, a grad student at CMU developed Lycos. First to use a standard IR system as developed for the DARPA Tipster project. First to index a large set of pages.
- In late 1995, DEC developed Altavista.
- In 1998, Larry Page and Sergey Brin, Ph.D. students at Stanford, started Google. Main advance is use of *link analysis* to rank results partially based on authority.

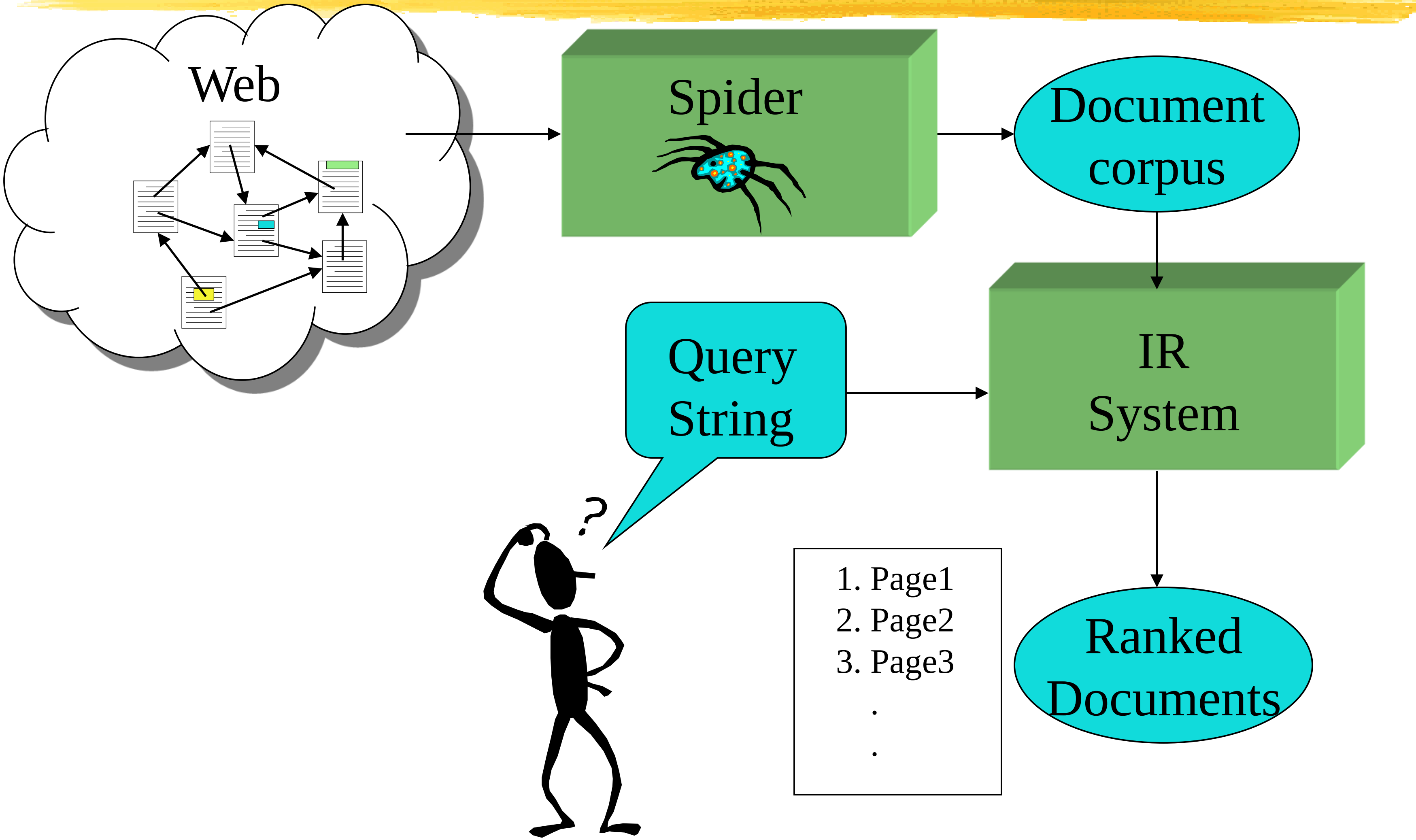
Web Challenges for IR

- Distributed Data: Documents spread over millions of different web servers.
- Volatile Data: Many documents change or disappear rapidly (e.g. dead links).
- Large Volume: Billions of separate documents.
- Unstructured and Redundant Data: No uniform structure, HTML errors, up to 30% (near) duplicate documents.
- Quality of Data: No editorial control, false information, poor quality writing, typos, etc.
- Heterogeneous Data: Multiple media types (images, video, VRML), languages, character sets, etc.

Zipf's Law on the Web

- Number of in-links/out-links to/from a page has a Zipfian distribution.
- Length of web pages has a Zipfian distribution.
- Number of hits to a web page has a Zipfian distribution.

Web Search Using IR



Search Engine Architecture

- Spider
  - Crawls the web to find pages. Follows hyperlinks. Never stops
- Indexer
  - Produces data structures for fast searching of all words in the pages
- Retriever
  - Query interface
  - Database lookup to find hits
  - Ranking

Crawlers (Spiders, Bots)

- Retrieve web pages for indexing by search engines
- Start with an initial page P_0. Find URLs on P_0 and add them to a queue
- When done with P_0, pass it to an indexing program, get a page P_1 from the queue and repeat

- Issues
  - Which page to look at next? (Special subjects, recency)
  - Avoid overloading a site
  - How deep within a site to go (drill-down)?
  - How frequently to visit pages?

Metasearchers

All the engines operate differently. Different

- sizes
- query languages
- crawling algorithms
- storage policies (stop words, punctuation, fonts)
- freshness
- ranking

Submit the same query to many engines and collect the results

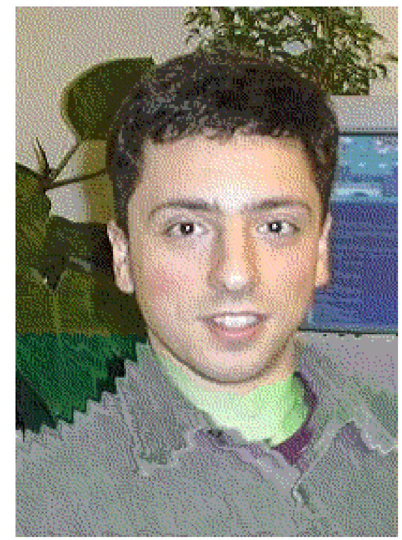
Google's PageRank Algorithm

Sergey Brin - Microsoft Internet Explorer

文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)

地址(D) E:\IRClass05\HITSRef\2\Sergey Brin.htm

Sergey Brin's Home Page



Ph.D. student in Computer Science at Stanford - mailto:sergey@cs.stanford.edu

Research

Currently I am at Google.

In fall '98 I taught CS 349.

Data Mining

A major research interest is data mining and I run a meeting group here at Stanford. For more information take a look at the MIDAS home page or see the datamine maling list achive. Here are some recent publications:

- **Extracting Patterns and Relations from the World Wide Web**

  by Sergey Brin.
  We demonstrate a technique for extracting relations from the WWW based on the duality of patterns and relations. We experiment with it by extracting a relations of books. WebDB Workshop at EDBT '98 (postscript).

- **Dynamic Data Mining: A New Architecture for Data with High Dimensionality**

  by Sergey Brin and Lawrence Page

- Beyond Market Baskets: Generalizing Association Rules to Correlations

  by Craig Silverstein, Rajeev Motwani, and Sergey Brin.
  Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 265-276, Tuscon, Arizona, May 13-15 1997. (abstract , gzipped ps, bibtex). Note a version of this paper has been submitted to Data Mining and Knowledge Discovery.

- For the above two papers, we used the following census data.

World Wide Web

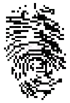
Research on the Web seems to be fashionable these days and I guess I'm no exception. Recently I have been working on the Google search engine with Larry Page.

- The Anatomy of a Large-Scale Hypertextual Web Search Engine

  by Lawrence Page and Sergey Brin.
  Submitted to WWW7.

- **PageRank, an Eigenvector based Ranking Approach for Hypertext**

  by Lawrence Page and Sergey Brin.
  This is work in progress which we intend to submit to SIGIR '98.



GNAT's

This project involved indexing multidimensional data for near-neighbor searches. The kind of applications I envision are identity comparisons, information finding, molecular biology, ...

- Near Neighbor Search in Large Metric Spaces

Lawrence or Larry Page's Page - Microsoft Internet Explorer

文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)

后退 搜索 收藏夹 媒体

地址(D) E:\IRClass05\HITSRef\2\Lawrence or Larry Page's Page.htm 转到 链接

Larry (Lawrence) Page
Ph.D. Student
Computer Science Department
Stanford University

Member of Terry Winograd's Project on People, Computers, and Design.

Currently working on Google, a search engine for the Web. Papers and a demo are available off this page.

email: page@cs.stanford.edu
office: Gates 360
phone: (650) 330-0100
mail: Department of Computer Science
Gates Building Room #360
Stanford University
Stanford, CA 94305-2140

Internet

Google's PageRank Algorithm

Link Structure of the Web

Assumption: A **link** in page A to page B is a **recommendation** of page B by the author of A (we say B is *successor* of A)

The "quality" of a page is related to the number of links that point to it (its in-degree)

Apply recursively: Quality of a page is related to

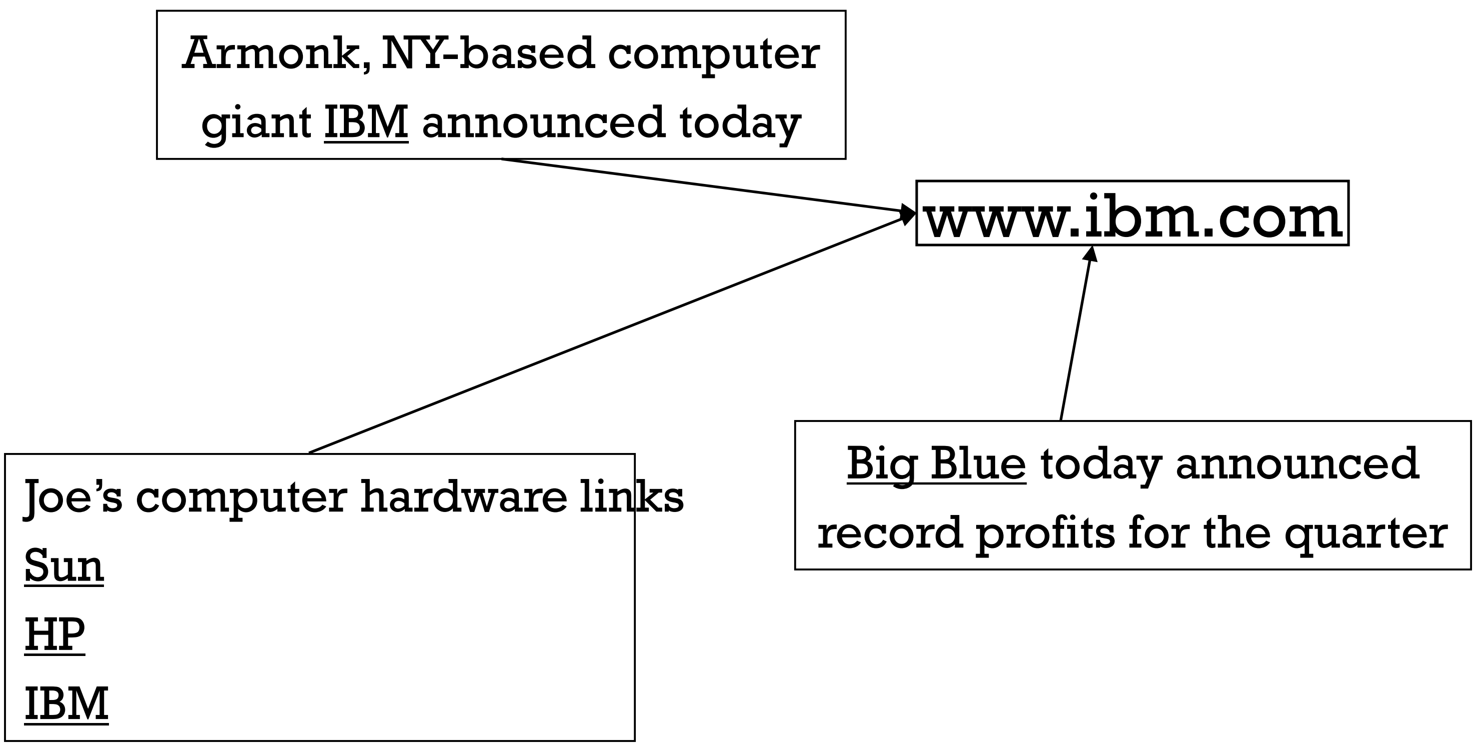
its in-degree, and to

the *quality* of pages linking to it

PageRank Algorithm (Brinn & Page, 1998)

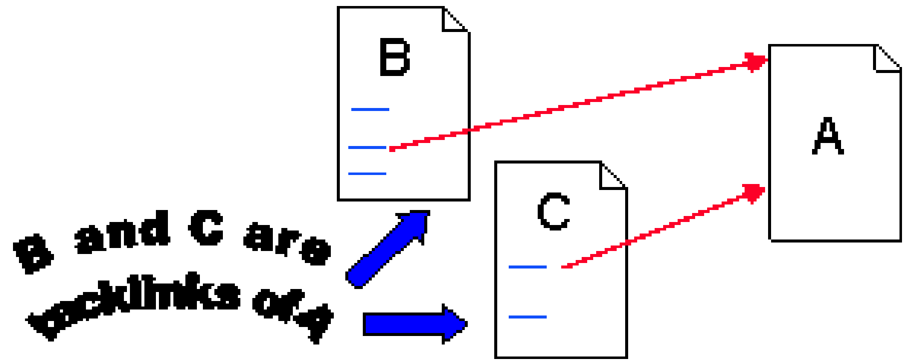
Google's PageRank Algorithm

* anchor



Google's PageRank Algorithm

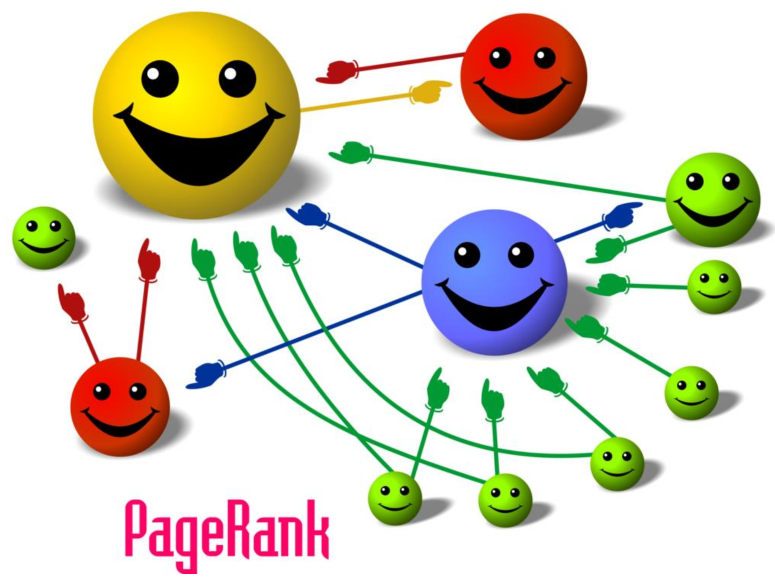
Backlink



$$\sum_{i\geq 1} x_i I_{\{i\to j\}} = \lambda x_j, \forall j.$$

$$xA = \lambda x,$$

x： 行向量



Formalizing Our Intuitive Notion of Webpage Importance.

- Let u be a webpage
- Let F_u be the set of pages u points to (forward links)
- Let B_u be the set of pages that point to u (backlinks)
- Let $N_u = |F_u|$ be the number of links from u
- Let r be a simple ranking function

$$r(u) = \sum_{v \in B_u} \frac{r(v)}{N_v}$$

Computing $r(u)$ Iteratively.

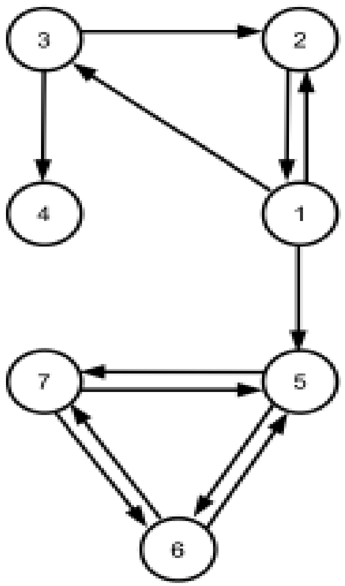
If n is the number of webpages, and we let

$$r_0(u) = 1/n$$

then

$$r_{k+1}(u) = \sum_{v \in B_u} \frac{r_k(v)}{N_v}$$

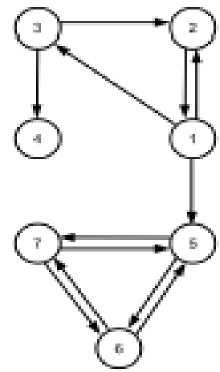
An Example of the Basic Ranking Function



example adapted from Langville and Meyer

initial	iteration$_1$	iteration$_2$	rank
$r_0(1) = .143$	$r_1(1) = .143$	$r_2(1) = .119$	4
$r_0(2) = .143$	$r_1(2) = .119$	$r_2(2) = .063$	5
$r_0(3) = .143$	$r_1(3) = .048$	$r_2(3) = .040$	6
$r_0(4) = .143$	$r_1(4) = .024$	$r_2(4) = .019$	7
$r_0(5) = .143$	$r_1(5) = .190$	$r_2(5) = .212$	1
$r_0(6) = .143$	$r_1(6) = .167$	$r_2(6) = .195$	3
$r_0(7) = .143$	$r_1(7) = .179$	$r_2(7) = .204$	2

A Matrix Representation of the Rank Formula.

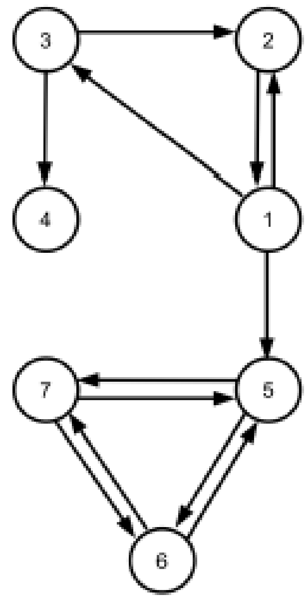


H: $n \times n$ hyperlink matrix
π^T: $1 \times n$ vector of rank values

$$\mathbf{H} = \begin{pmatrix} 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 \end{pmatrix}$$

Basic rank formula:
$\pi^{(k+1)T} = \pi^{(k)T} H$

Some Problems



- the subgraph of u_5, u_6, u_7 forms a rank sink.
- u_4 is a dangling node. Dangling nodes result in $\mathbf{0}^T$ rows in $\mathbf{H}$.
- will this process in general converge? converge to a unique ranking? does convergence depend on the starting vector $\pi^{(0)T}$?

Fix for Dangling Nodes

Motivation: *random surfer model*

1.) Replace $\mathbf{0}^T$ rows of $\mathbf{H}$ with $1/n\, \mathbf{e}^T$—>$\mathbf{H}$ is now stochastic.

$\mathbf{S} = \mathbf{H} + 1/n\, \mathbf{a}\, \mathbf{e}^T$

$$\mathbf{S} = \begin{pmatrix} 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 \end{pmatrix}$$

Fix for Rank Sinks

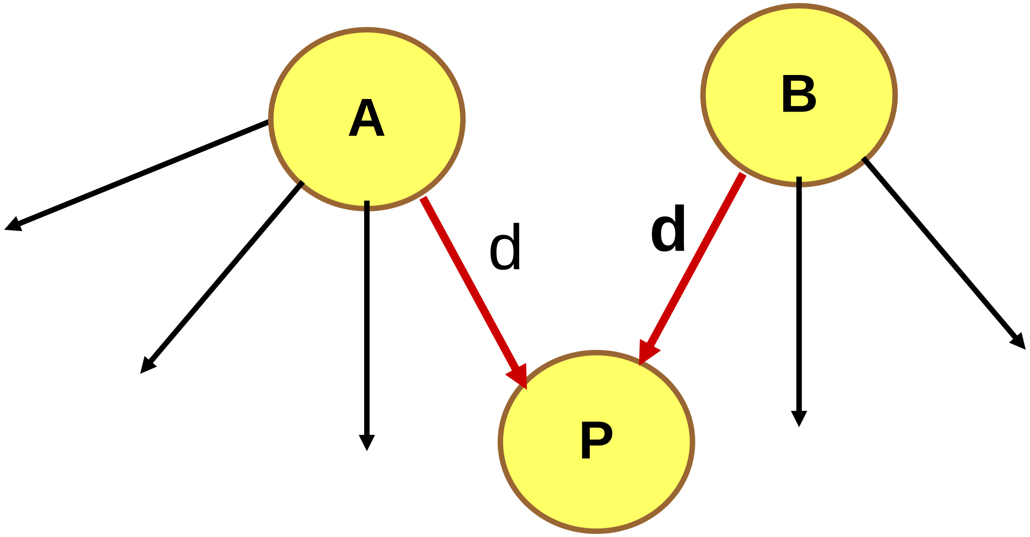
2.) Define $\alpha \in (0, 1)$ as a teleportation parameter.

This gives us the Google matrix:

$$\mathbf{G} = \alpha \mathbf{S} + (1 - \alpha) 1/n \, \mathbf{e} \, \mathbf{e}^T$$

Note: Google initially chose $\alpha = .85$.

PageRank Example



PageRank of P is

d∗[(PageRank of A)/4 + (PageRank of B)/3)] +(1-d)/n

Properties of Google matrix.

$$\mathbf{G} = \alpha \mathbf{S} + (1-\alpha) 1/n\, \mathbf{e}\, \mathbf{e}^T$$

G is:

- stochastic
- irreducible
- aperiodic
- primitive
- dense

$$\begin{aligned}
\mathbf{G} &= \alpha \mathbf{S} + (1-\alpha) 1/n\, \mathbf{e}\, \mathbf{e}^T && (1)\\
&= \alpha(\mathbf{H} + 1/n\, \mathbf{a}\, \mathbf{e}^T) + (1-\alpha) 1/n\, \mathbf{e}\, \mathbf{e}^T && (2)\\
&= \alpha \mathbf{H} + (\alpha \mathbf{a} + (1-\alpha)\, \mathbf{e}) 1/n\, \mathbf{e}^T && (3)
\end{aligned}$$

The PageRank Equation

$$\pi^{(k+1)T} = \pi^{(k)T}\mathbf{G}$$

$$\begin{aligned}
\mathbf{G} &= \alpha\mathbf{S} + (1-\alpha)1/n\,\mathbf{e}\,\mathbf{e}^T && (4)\\
&= \alpha(\mathbf{H} + 1/n\mathbf{a}\mathbf{e}^T) + (1-\alpha)1/n\,\mathbf{e}\,\mathbf{e}^T && (5)\\
&= \alpha\mathbf{H} + (\alpha\mathbf{a} + (1-\alpha)\,\mathbf{e})1/n\,\mathbf{e}^T && (6)
\end{aligned}$$

- **H**: sparse hyperlink matrix
- **S**: sparse stochastic matrix
- **G**: dense stochastic, primitive matrix
- **E**: dense teleportation matrix
- n: number of pages
- α: scaling parameter
- π^T: stationary PageRank vector
- $\mathbf{a}^T$: binary dangling node vector

Example PageRank Calculation

$$\mathbf{G} = \begin{pmatrix} 0.021 & 0.305 & 0.305 & 0.021 & 0.305 & 0.021 & 0.021 \\ 0.871 & 0.021 & 0.021 & 0.021 & 0.021 & 0.021 & 0.021 \\ 0.021 & 0.446 & 0.021 & 0.446 & 0.021 & 0.021 & 0.021 \\ 0.143 & 0.143 & 0.143 & 0.143 & 0.143 & 0.143 & 0.143 \\ 0.021 & 0.021 & 0.021 & 0.021 & 0.021 & 0.446 & 0.446 \\ 0.021 & 0.021 & 0.021 & 0.021 & 0.446 & 0.021 & 0.446 \\ 0.021 & 0.021 & 0.021 & 0.021 & 0.446 & 0.446 & 0.021 \end{pmatrix}$$

$$\pi^T = (0.093, 0.077, 0.054, 0.050, 0.254, 0.236, 0.236)$$

ranks: $(u_5 u_6 u_7 u_1 u_2 u_3 u_4)$

Regardless of the method for filling in and storing the entries of $\bar{\bar{P}}$, PageRank is determined by computing the stationary solution π^T of the Markov chain. The row vector π^T can be found by solving either the eigenvector problem

$$\pi^T \bar{\bar{P}} = \pi^T,$$

or by solving the homogeneous linear system

$$\pi^T (I - \bar{\bar{P}}) = 0^T,$$

where I is the identity matrix. Both formulations are subject to an additional equation, the normalization equation $\pi^T e = 1$, where e is the column vector of all 1's. The normalization equation insures that π^T is a probability vector. The i^{th} element of π^T, π_i, is the PageRank of page i. Stewart's book, "An Introduction to the Numerical Solution of Markov Chains" [108], contains an excellent presentation of the various methods of solving the Markov chain problem.

Google's PageRank Algorithm

[1]A matrix is *irreducible* if its graph shows that every node is reachable from every other node. A nonnegative, irreducible matrix is *primitive* if it has only one eigenvalue on its spectral circle. An irreducible Markov chain with a primitive transition matrix is called an aperiodic chain. Frobenius discovered a simple test for primitivity: the matrix $\mathbf{A} \geq 0$ is primitive if and only if $\mathbf{A}^m > 0$ for some $m > 0$ [89]. This test is useful in determining whether the power method applied to a matrix will converge.

PERRON-FROBENIUS THEOREM FOR PRIMITIVE MATRICES

*If A is an **nxn** nonnegative primitive matrix then*

1. one of its eigenvalues is positive and greater than (in absolute value) all other eigenvalues

2. there is a positive eigenvector corresponding to that eigenvalue

α

$0 < \alpha < 1$

$$\mathbf{G} = \alpha\mathbf{S} + (1-\alpha)1/n\,\mathbf{e}\,\mathbf{e}^T$$

- hyperlink structure vs teleportation matrix **E**
- speed of eigenvector convergence: $-\tau/log_{10}\alpha$ iterations
- $\alpha \to 1$: more sensitive to structure changes; slower convergence
- $\alpha \to 0$: more insensitive to structure changes; faster convergence.

E

- democratic: $\mathbf{G} = \alpha \mathbf{S} + (1 - \alpha) 1/n \; \mathbf{e} \, \mathbf{e}^T$
- personal: $\mathbf{G} = \alpha \mathbf{S} + (1 - \alpha) \, \mathbf{e} \, \mathbf{v}^T$

PageRank Example

PR(A)=0.5+0.5 PR(C)

PR(B)=0.5+0.5 (PR(A)/2)

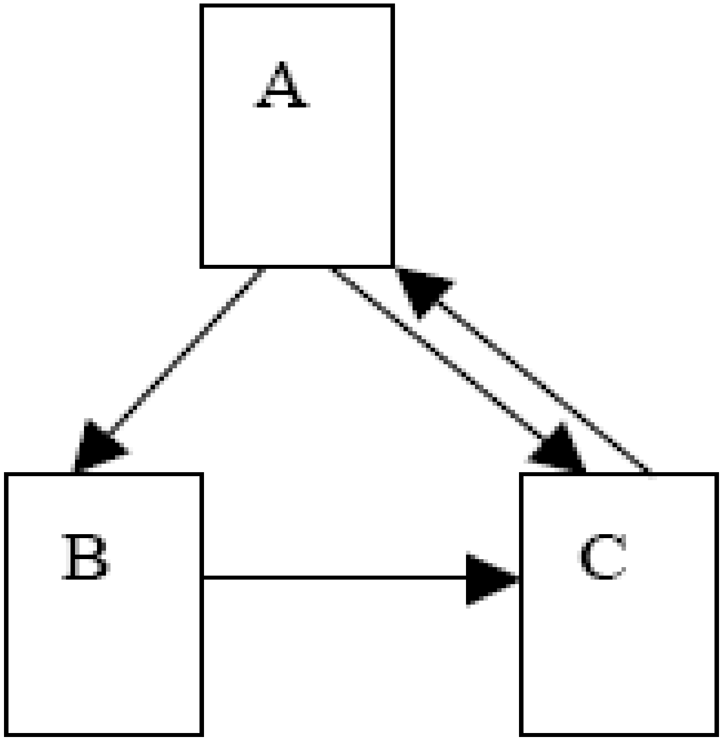
PR(C)=0.5+0.5 (PR(A)/2+ PR(B))

解方程，得：

PR(A)=14/13=1.0769

PR(B)=10/13=0.76923

PR(C)=15/13=1.1538



PageRank Algorithm

初值选为 1

迭代次数	PR(A)	PR(B)	PR(C)
0	1	1	1
1	1	0.75	1.125
2	1.0625	0.76563	1.1484
3	1.0742	0.76855	1.1528
4	1.0764	0.7691	1.1537
5	1.0768	0.76921	1.1538
6	1.0769	0.76923	1.1538
7	1.0769	0.76923	1.1538

PageRank Algorithm

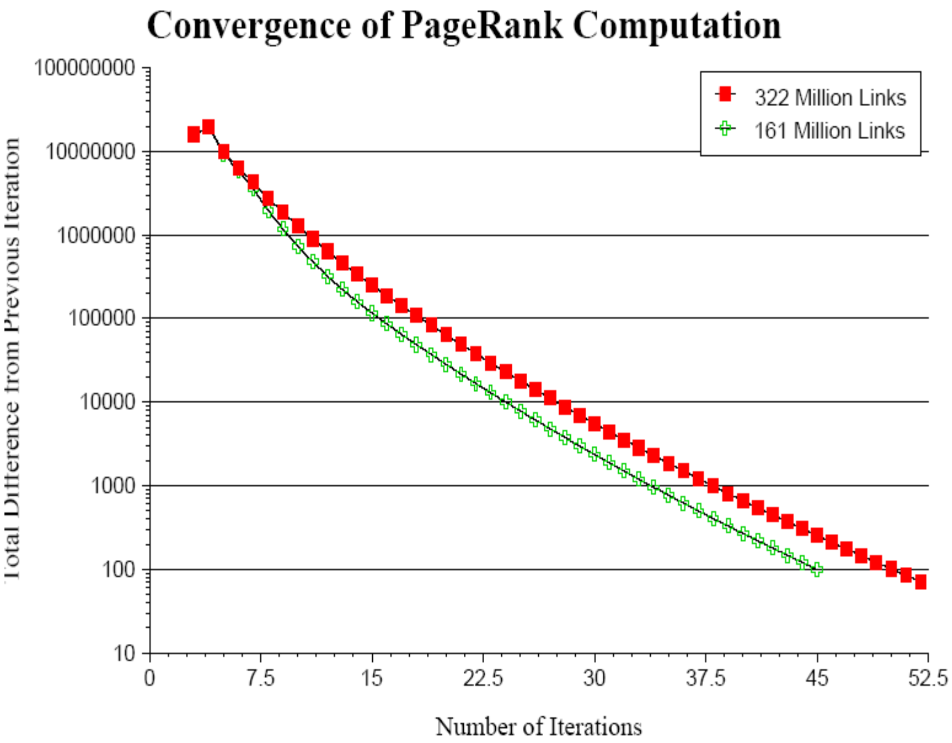
初值选为 1.5

迭代次数	PR(A)	PR(B)	PR(C)
0	1.5	1.5	1.5
1	1.25	0.8125	1.2188
2	1.1094	0.77734	1.166
3	1.083	0.77075	1.1561
4	1.0781	0.76952	1.1543
5	1.0771	0.76928	1.1539
6	1.077	0.76924	1.1539
7	1.0769	0.76923	1.1538
8	1.0769	0.76923	1.1538

Some PageRank Results

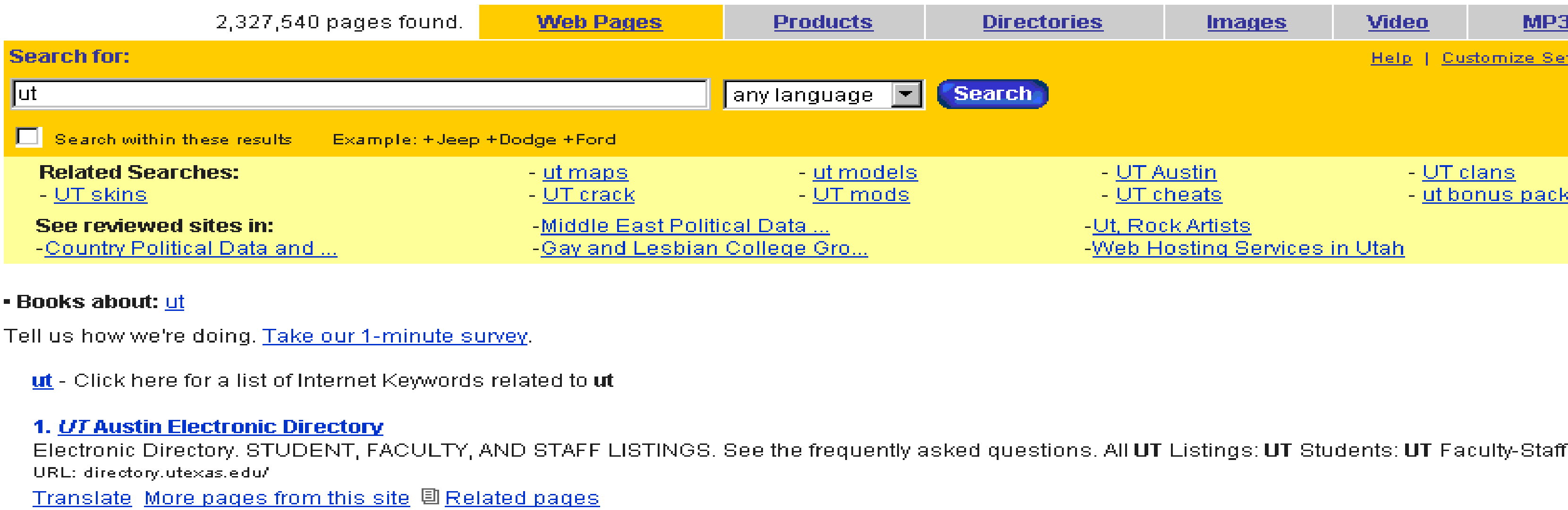
Size	Iterations
1000	4
2000	5
4000	5
8000	5
16000	6

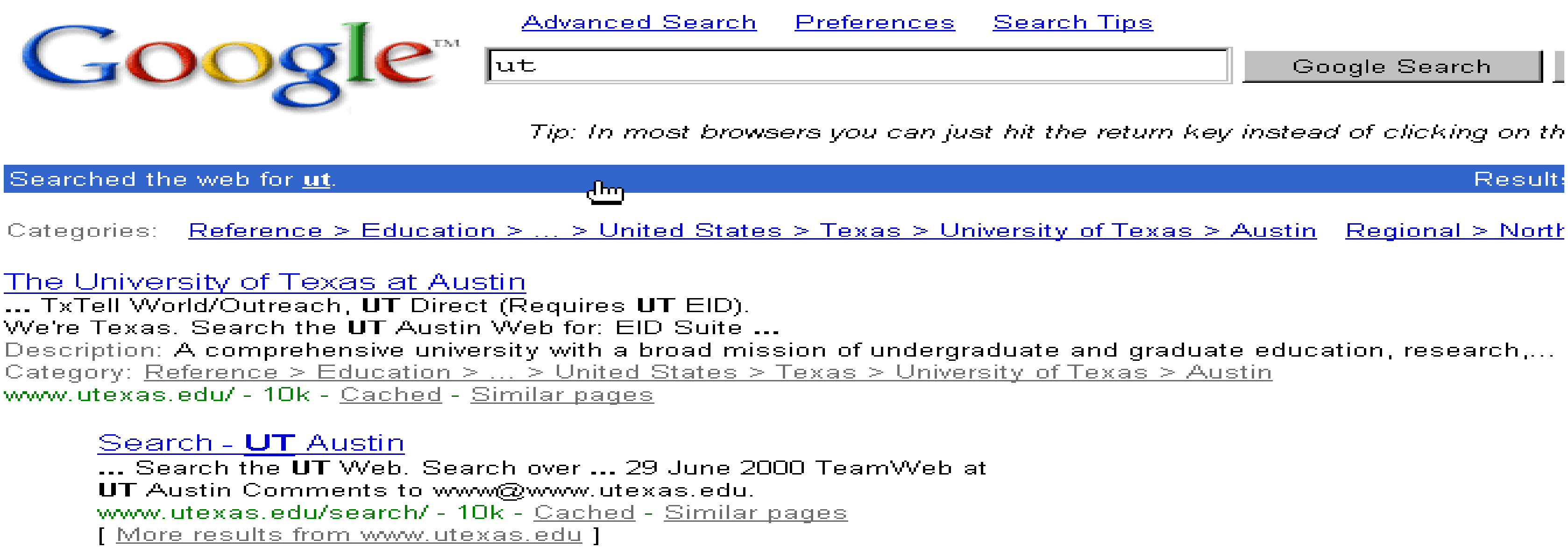
PageRank Convergence



- Needs to converge to PageRank in as few iterations as possible
- PageRank usually converges within 100 iterations
- PageRank is stable

PageRank Evaluation





PageRank Evaluation

24 pages found. | **Web Pages** | **Products** | **Directories** | **Images** | **Video** | **MP3/Auc**

Search for: Help | Customize Settin

yuqiang guan — any language — **Search**

Search within these results Tip: Use a plus sign to require a keyword.

Tell us how we're doing. Take our 1-minute survey.

guan - Click here for a list of Internet Keywords related to **guan**

1. 1999 Combinatorics, Graph Theory and Computing Conference
1999 Combinatorics, Graph Theory. and Computing Conference. Monday. Tuesday. Wednesday. Thursday. Friday. Authors. Main Page. Invited talks. Martin...
URL: www.math.fau.edu/locke/99CGTC07.htm
Translate More pages from this site Related pages

Advanced Search Preferences Search Tips

yuqiang guan — Google Search

Searched the web for **yuqiang guan**.

Guan's magic galaxy!
... Magic lyrics. Magic Guestbook Sign View. Last update in Mar. 2000, by **Yuqiang Guan** yguan@cs.utexas.edu.
www.cs.utexas.edu/users/yguan/ - 4k - Cached - Similar pages

UTCS Rwho/Finger
UTCS Rwho/Finger: yguan. Check the UT Directory for the name "**Yuqiang Guan**" Finger another name **...**
www.cs.utexas.edu/cgi/rwhofinger.cgi/yguan - 3k - Cached - Similar pages
[More results from www.cs.utexas.edu]

PageRank Issues

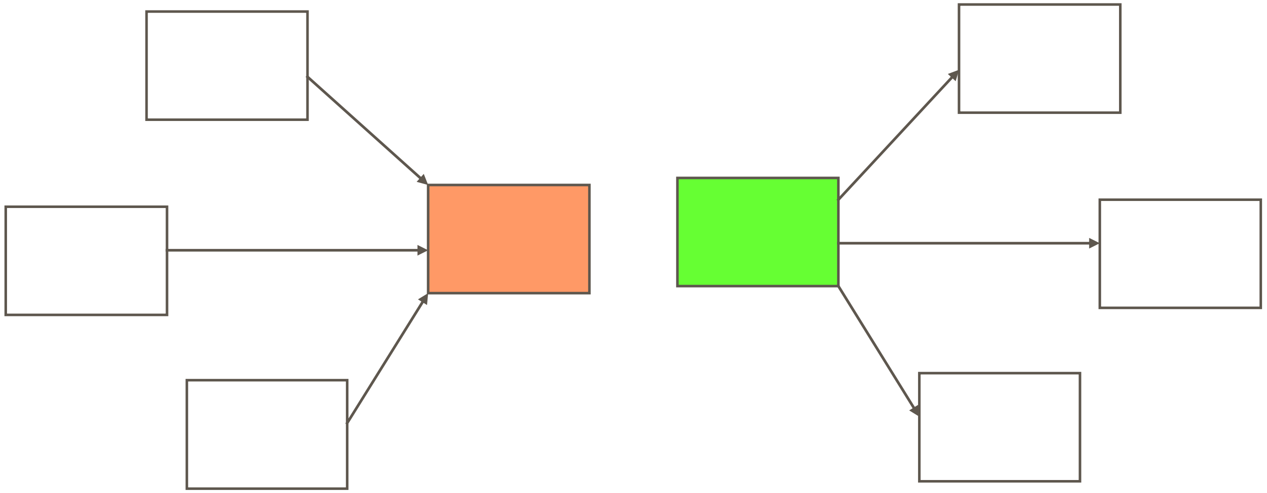
* Users are no random walkers
  – Content based methods
* Reinforcing effects/bias towards main pages
* Linkage spam
  – PageRank favors pages that managed to get other pages to link to them
  – Linkage not necessarily a sign of relevancy, only of promotion (advertisement…)
* No query specific rank

HITS: Intuition

J. Kleinberg, 1998

Authority **comes from** in-edges.

hub **comes from** out-edges.



Jon Kleinberg's Homepage - Microsoft Internet Explorer

文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)

后退 搜索 收藏夹 媒体

地址(D) E:\IRClass05\HITSRef\3\Jon Kleinberg's Homepage.htm 转到 链接

Jon Kleinberg

Professor of Computer Science
Cornell University
Ithaca, NY 14853

My research is concerned with algorithms that exploit the combinatorial structure of networks and information. My recent work has included

- link analysis and modeling of the World Wide Web and related information networks;
- discrete optimization and network algorithms; and
- algorithmic approaches to clustering, indexing, and data mining.

My work has been supported by an NSF Career Award, an ONR Young Investigator Award, an Alfred P. Sloan Foundation Fellowship, a David and Lucile Packard Foundation Fellowship, and grants from the NSF.

I am currently the program chair for STOC 2006, the ACM Symposium on Theory of Computing.

Current Ph.D. students

- Mark Sandler (graduating Spring 2006).
- Alex Slivkins (graduating Spring 2006).

Publications

Textbook on Algorithms

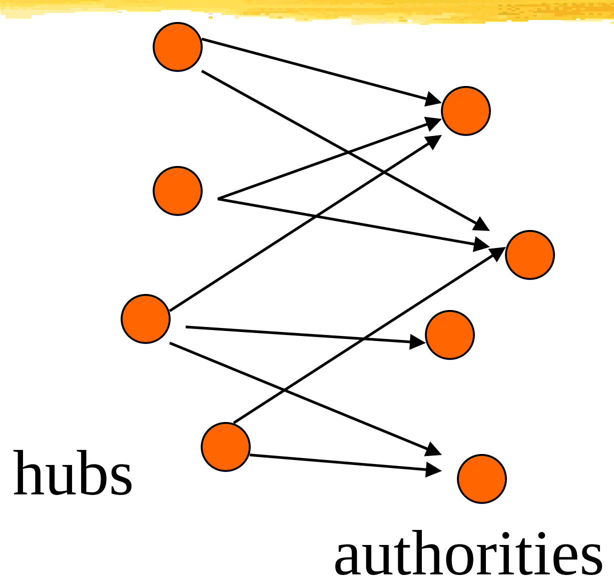
- J. Kleinberg, E. Tardos. Algorithm Design. Addison-Wesley, 2005.

Research Papers: Background

- The papers below are also available in a chronological list. Additional bibliographic information can be found at

完毕 Internet

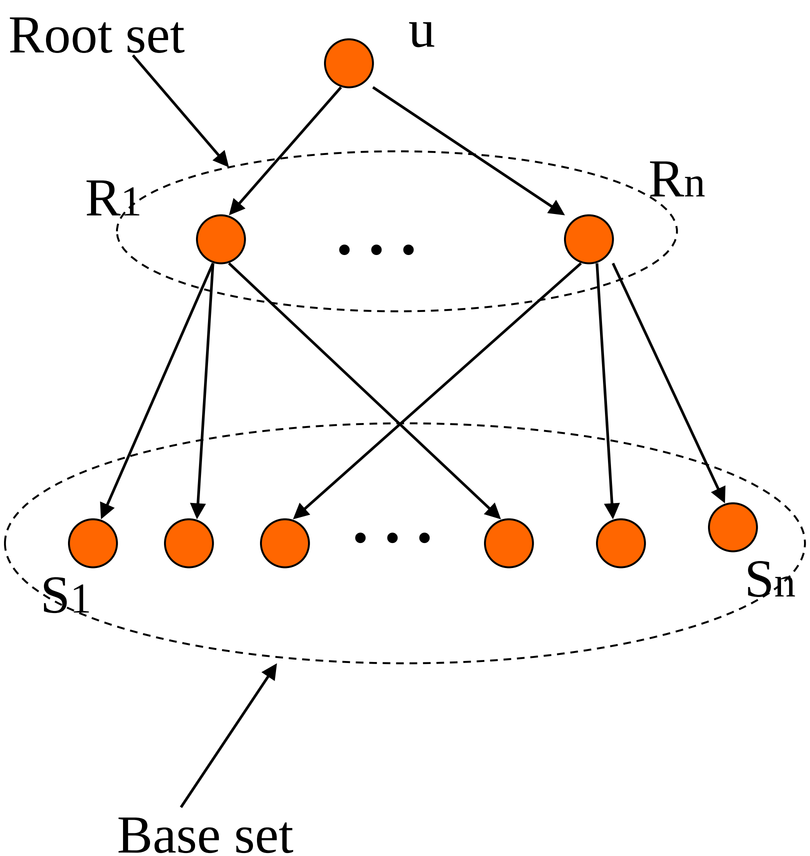
Authorities and Hubs



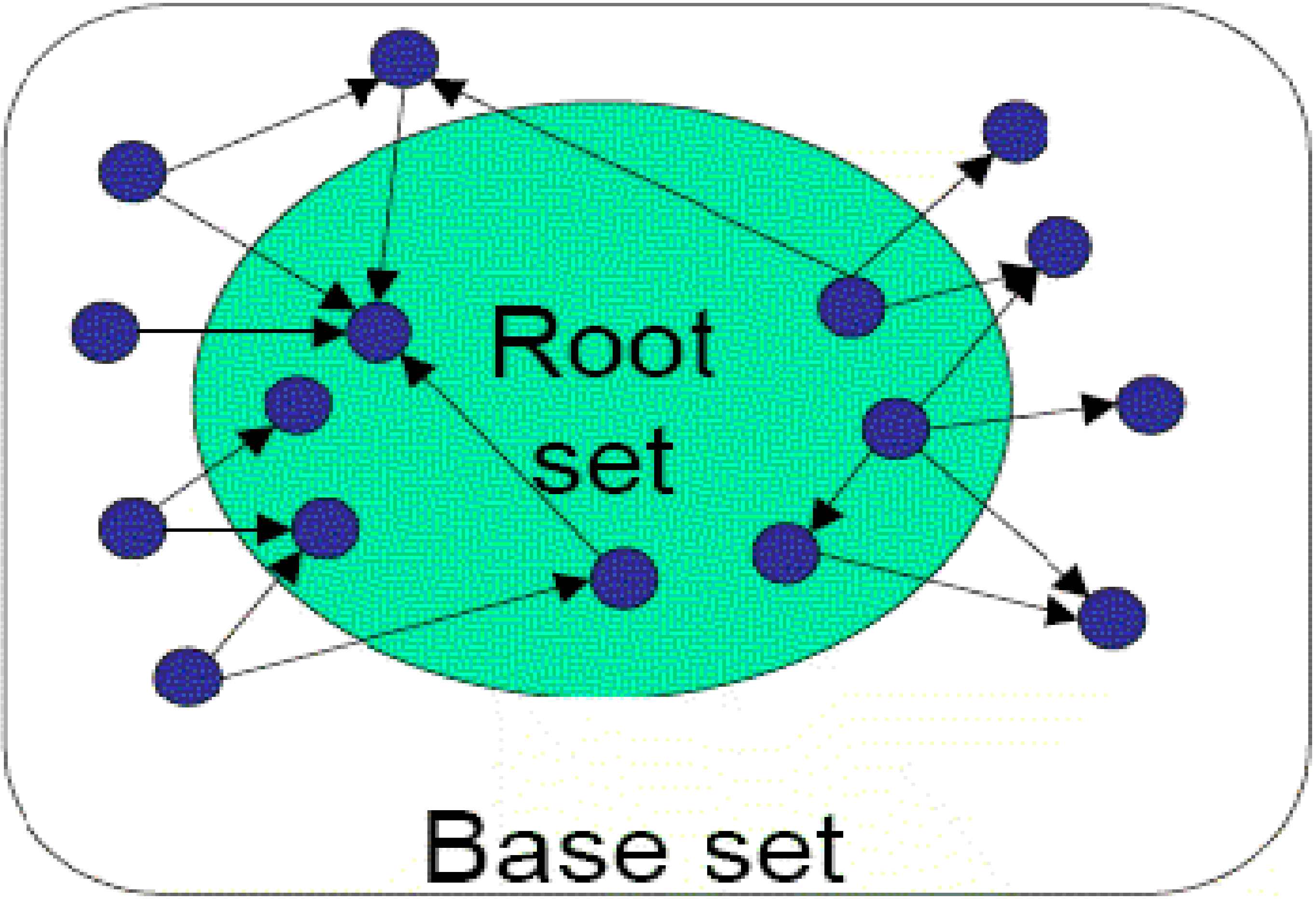
- **A good authority is a page that is pointed by many good hubs, while a good hub is a page that points to many good authorities.**
- **This is the *mutually reinforcing relationship*.**

HITS (Hyperlink-Induced Topic Search)

- The focused subgraph is created by first taking the highest-ranked pages from a text-based search engine as a *root set* R.
- R is expanded into the *base set* S by taking all sites pointing to or pointed at by a site in R.
- Note that while R may fail to contain some “important” authorities, S will probably contain them.



HITS (Hyperlink-Induced Topic Search)



HITS Algorithm

Depended on a search engine

For each node u in the graph calculated Authorities scores (a_u) and Hubs scores (h_u):

Initialize $h_u = a_u = 1$

Repeat until convergence:

$$a_{\mathrm{u}} := \sum_{\mathrm{v} \to \mathrm{u}} h_{\mathrm{v}} \quad and \quad h_u := \sum_{u \to v} a_v$$

$\sum_u h_u \quad and \quad \sum_v a_v$ are normalized to 1

HITS Algorithm

initialize authority and hub weights, x_0 and y_0

while (not converged)

  for each vertex i

$$x_{k+1}(i) = \sum_{j \in B_i} y_k(j)$$

$$y_{k+1}(i) = \sum_{j \in F_i} x_k(j)$$

  end

End

HITS is stable.

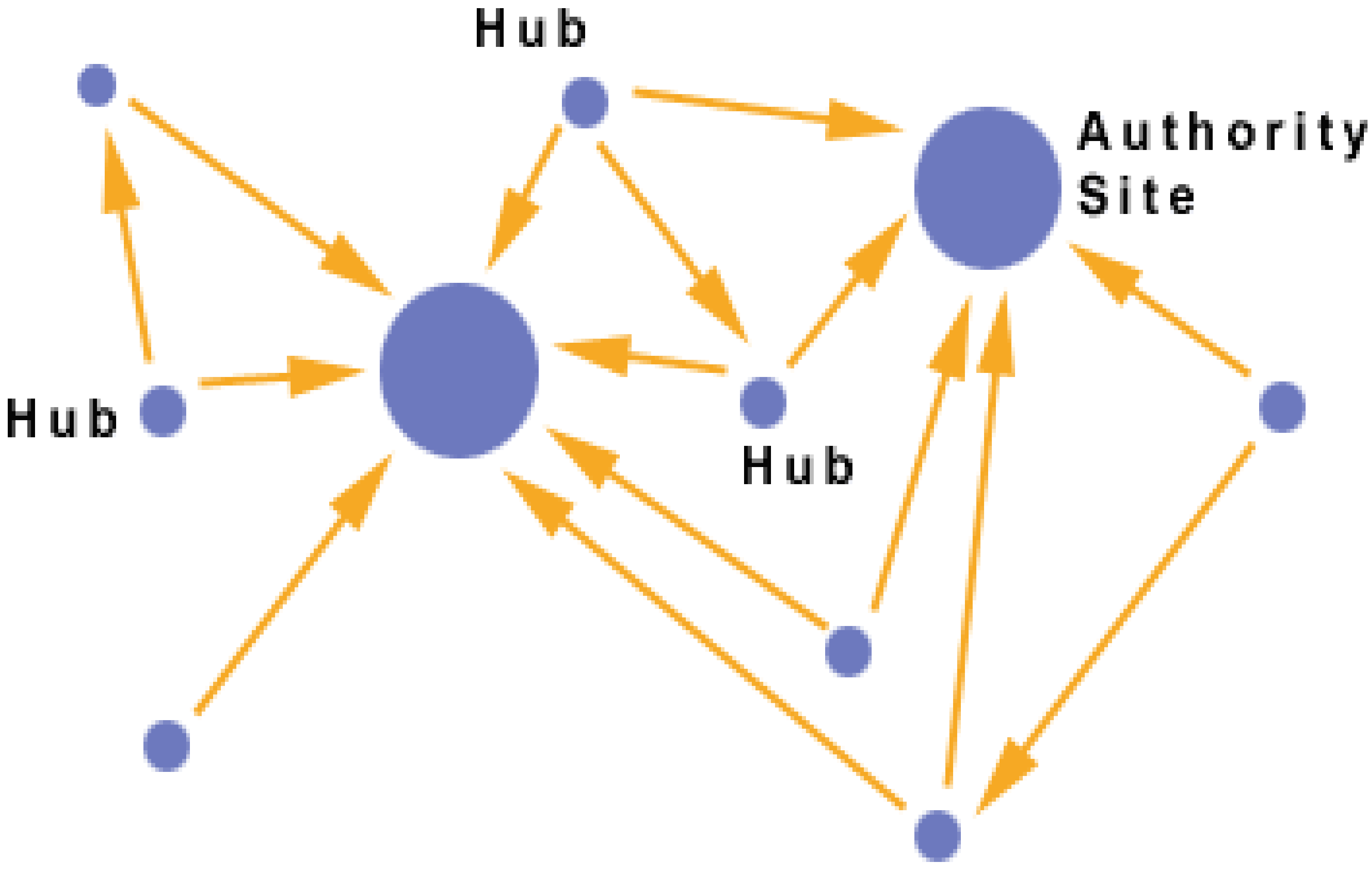
这样的递归式也容易用矩阵方法表示。令所有选出来的网页都进行标号，我们得到所有网页的编号集{1,2,…,n}。令相邻矩阵A 为一个 $n \times n$ 的矩阵，如果存在一个从网页i链接到网页j 的超链，就令矩阵中的第（i,j）个元素置为 1，其它各项置为 0。同时，我们将所有网页的权威型权值x和目录型权值y都表示成向量形式 $x = (x_1, x_2, \dots x_n)$，$y = (y_1, y_2, \dots y_n)$。由此我们可以得到计算x和y的简单矩阵公式：$y = A \cdot x$， $x = A^T \cdot y$，其中 A^T 是A的转置矩阵。进一步，我们有：

$$x = A^T \cdot y = A^T Ax = (A^T A)x$$

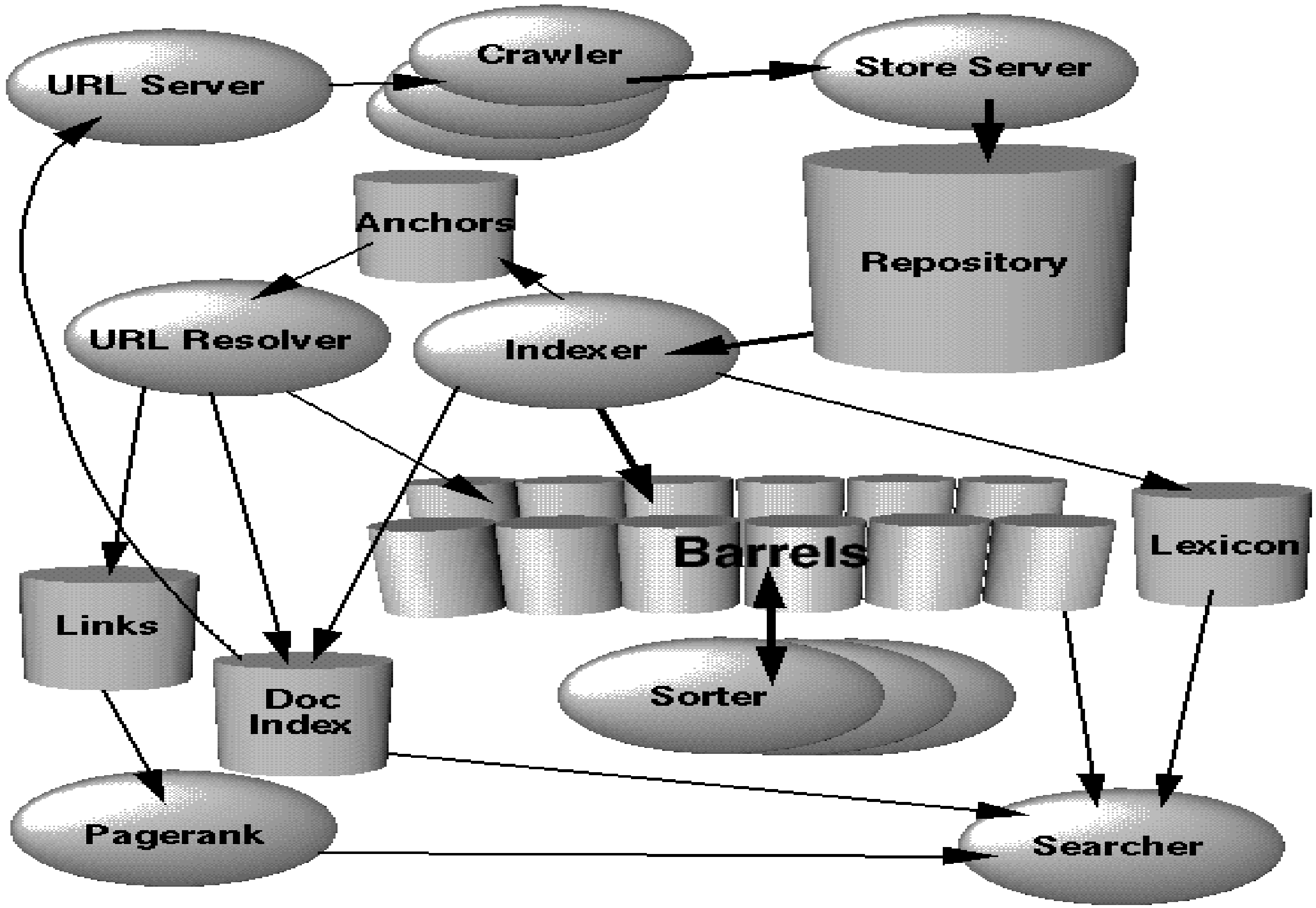
$$y = A \cdot x = AA^T y = (AA^T)y$$

经过一定次数的递归运算后，会得到集合中每个网页的权威型权值和目录型权值。按照这两个不同的权值，分别取出前 k 个返回给用户。

根据 Clever 系统自己的测试数据，对于返回给用户的前 10 个检索结果，Clever 系统在 50%的情况下获得了高于 Yahoo!和 AltaVista 的用户评价。


Hub
Authority
Site
Hub
Hub

Google Architecture Overview



Simple Structure of Google Search Engine