

# COL106

# Data Structures and Algorithms

Subodh Sharma and Rahul Garg

# Applications of Graphs: Search Results Ranking

# References

1. Kleinberg, Jon M. "[Authoritative sources in a hyperlinked environment](#)." SODA. Vol. 98. 1998.
2. Kleinberg, J. M., Kumar, R., Raghavan, P., Rajagopalan, S., & Tomkins, A. S. (1999). [The web as a graph: Measurements, models, and methods](#). In Computing and Combinatorics: 5th Annual International Conference, COCOON'99 Tokyo, Japan, July 26–28, 1999 Proceedings 5 (pp. 1-17). Springer Berlin Heidelberg.
3. Brin, Sergey, and Lawrence Page. "[The anatomy of a large-scale hypertextual web search engine](#)." Computer networks and ISDN systems 30.1-7 (1998): 107-117.
4. [The PageRank Citation Ranking: Bringing Order to the Web](#). by Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. Technical Report SIDL-WP-1999-0120, Stanford Digital Library Technologies Project

# A Brief History of the Internet

- 1969 – ARPANET project
  - Advanced Research Projects Agency Network (ARPANET)
  - Advanced Research Projects Agency (ARPA) of the United States Department of Defense
- 1981 – expanded into CSNET
  - National Science Foundation (NSF) funded
  - Computer Science Network (CSNET)
- 1985 – NSFNET
  - Internet backbone for government agencies and universities
- 1990 – ARPANET project formally decommissioned
- 1995 – Backbone of the NSFNET decommissioned
  - Growth of commercial Internet
  - Growth of www and web pages

# Internet Search

- 1995 – 1998 growth of www
- Proliferation of web sites
- Growth of search engines
- Text based searching
- Find the most relevant pages that match the search query
  - Automated data driven approaches (altavista.com)
  - Manual indexing (yahoo.com)
- Poor search quality

# Search Results Ranking Problem

- Billions of web pages
- Millions of queries every day
- Millions of pages matching the query
- Can only show a few results on a page
- Which results to show?
- How to show the most relevant results on top?
- How to rank the search results in decreasing order of relevance?

# Modeling Web as a Graph

- From a large collection of “relevant” pages, find the most “authorities” pages
- Web as a graph
  - Vertices web-pages
  - Edges are the hyperlinks
- $G = (V, E)$  – Web graph
- $V$  = set of web pages
- $(u, v)$  in  $E$  if the page  $u$  points to the page  $v$

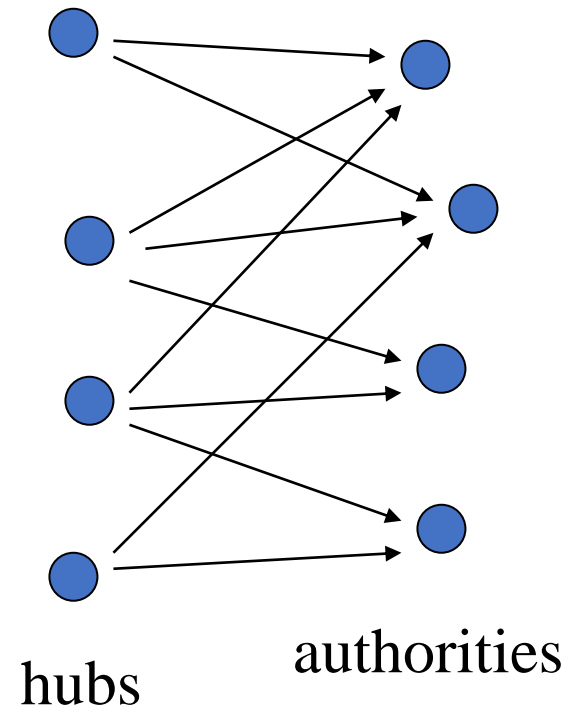
# Authoritative Sources in a Hyperlinked Environment (1998)

- Key ideas:
  - If a page  $u$  points to the page  $v$  (or  $(u, v) \in E$ ), then  $u$  contributes some authority to the page  $v$
  - Especially if  $u$  and  $v$  belong to different domains
  - E.g, COL106 home page pointing to a Textbook page
- Limitations
  - Navigational links or advertisement links do not confer authority – Remove links within the same domain
  - Authority pages may not point to other authority pages
  - Hub-pages may point to multiple authorities but not to other hubs
  - E.g., a page listing all the car dealers in the vicinity



# Bipartite Structure

- Top **authorities** will be pointed by a lot of **hubs**
- Top **hubs** will point to a lot of top **authorities**
- Can we **iterate** over this graph to find the **authority score** and the **hub score** for all the vertices?



# Iterative Algorithm

- Each page  $p$  is assigned two non-negative weights, an authority weight  $x$  and a hub weight  $y$ .
- Update the weights of  $x$  and  $y$

Authority Weight:  $x^{(p)} \leftarrow \sum_{q:(q,p) \in E} y^{(q)}$   
I Operation

Hub Weight :  $y^{(p)} \leftarrow \sum_{q:(p,q) \in E} x^{(q)}$   
O Operation

These operations add the weights of hubs into the authority weight and add the authority weights into the hub weight, respectively. Alternating these two operations will eventually result in an equilibrium value, or weight, for each page.

# Using Matrix Representation

*Definition:* Adjacency matrix of a graph  $G = (V, E)$  is a  $n \times n$  binary matrix  $A$  ( $n = |V|$ ) such that

$$a_{uv} = 1 \text{ if } (u, v) \in E$$

Authority Weight:  
I Operation

$$x^{(p)} \leftarrow \sum_{q:(q,p) \in E} y^{(q)} \quad x = A^T y$$

Hub Weight :  
O Operation

$$y^{(p)} \leftarrow \sum_{q:(p,q) \in E} x^{(q)} \quad y = Ax$$

# Convergence

- Iteration algorithm converges as  $k$  increases. That is, the weights (vectors) converge.

Let  $G = (V, E)$  with  $V = \{p_1, p_2 \dots p_n\}$

Let  $A$  be the adjacency matrix of  $G$ .

I and O operations can be written as  $x \leftarrow A^T y$      $y \leftarrow Ax$

Let  $x_i$  be the authority scores after  $i$  iterations.

Let  $y_i$  be the hub scores after  $i$  iterations.

Operation I     $x_i = A^T y_{i-1}$      $x_i = A^T A x_{i-1}$      $x_i = (A^T A)^i x_0$

Operation O     $y_i = A x_i$      $y_i = A A^T y_{i-1}$      $y_i = (A A^T)^i y_0$

# Eigenvalues and Eigenvectors

- For a square matrix  $A$ ,  $v$  is an eigenvector with an eigen value  $\lambda$  iff

$$Av = \lambda v$$

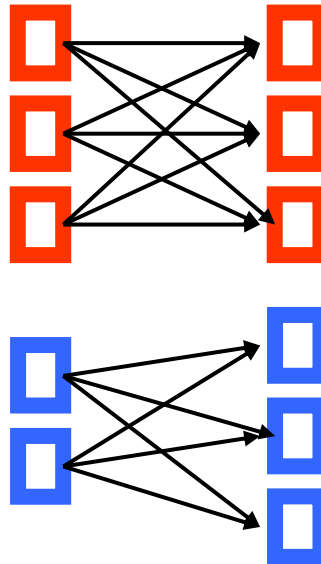
- For a  $n \times n$  matrix  $A$ , there are generally  $n$  eigenvectors and  $n$  eigen values
- $v$  is called the principal eigenvector if corresponding  $\lambda$  is the largest

# Convergence

- The vectors  $x_i$  and  $y_i$  converge to  $x^*$  and  $y^*$  respectively, where  $x^*$  and  $y^*$  are the principal Eigen vectors of  $A^T A$  and  $A A^T$
- Kleinberg says that 20 iterations are sufficient to obtain convergence
- The “principal eigenvector” represents the densest cluster in the focused subgraph
- The non-principal eigenvectors represent less dense areas in the subgraph

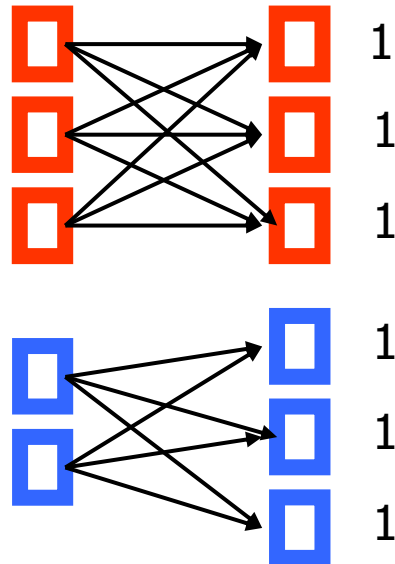
# Eigenvectors and Tightly Knit Communities

- The algorithm favors **dense community** of hubs and authorities



# Eigenvectors and Tightly Knit Communities

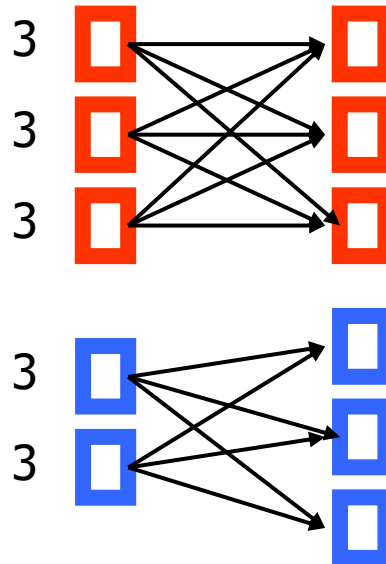
- The algorithm favors **dense community** of hubs and authorities





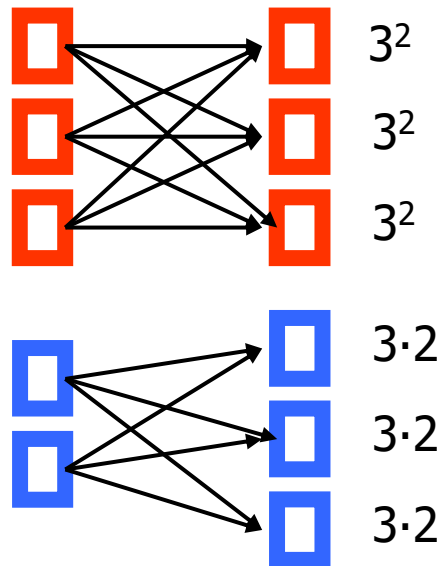
# Eigenvectors and Tightly Knit Communities

- The algorithm favors **dense community** of hubs and authorities



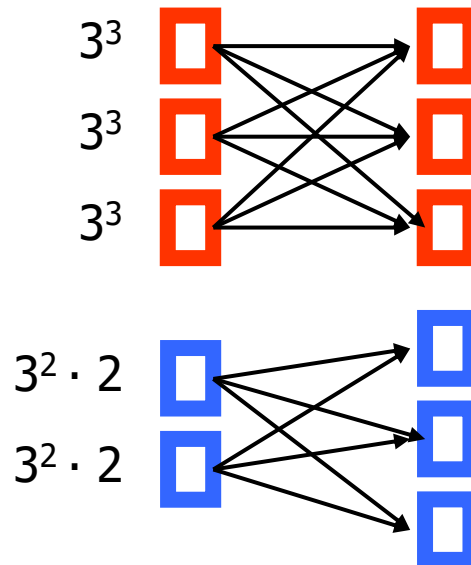
# Eigenvectors and Tightly Knit Communities

- The algorithm favors **dense community** of hubs and authorities



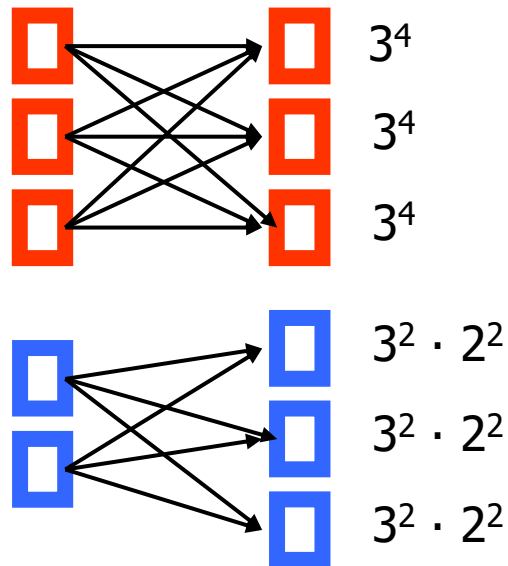
# Eigenvectors and Tightly Knit Communities

- The algorithm favors **dense community** of hubs and authorities



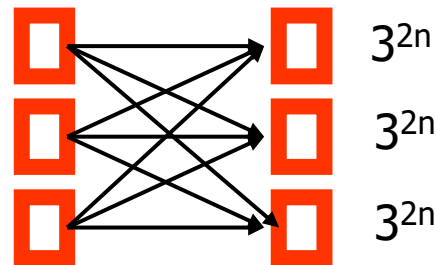
# Eigenvectors and Tightly Knit Communities

- The algorithm favors **dense community** of hubs and authorities



# Eigenvectors and Tightly Knit Communities

- The algorithm favors **dense community** of hubs and authorities

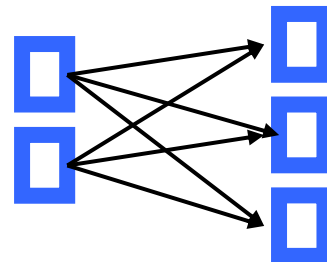


$3^{2n}$

$3^{2n}$

$3^{2n}$

after n iterations



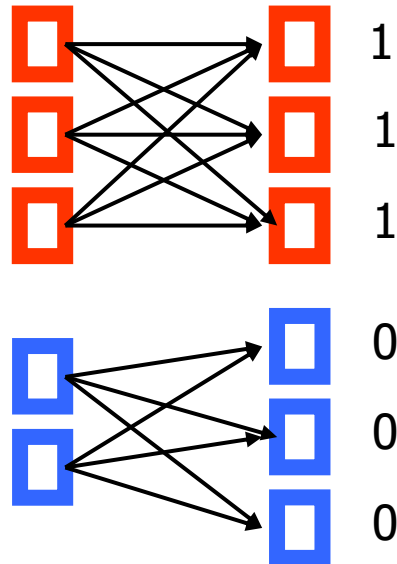
$3^n \cdot 2^n$

$3^n \cdot 2^n$

$3^n \cdot 2^n$

# Eigenvectors and Tightly Knit Communities

- The algorithm favors **dense community** of hubs and authorities

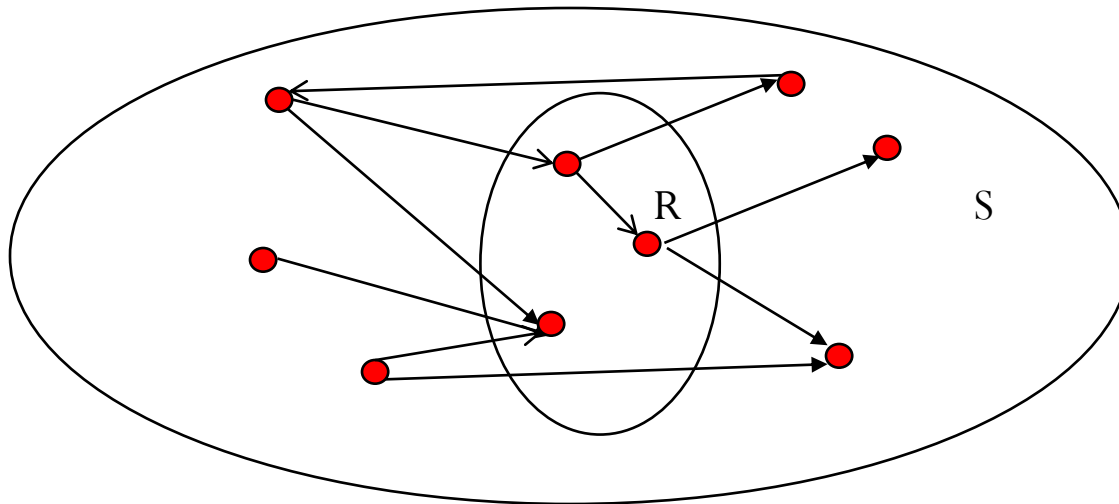


after normalization  
with the max  
element as  $n \rightarrow \infty$

# Hyperlink-Induced Topic Search (HITS) Algorithm

- First determines a set of relevant pages for the query called the *base* set  $S$
- Analyze the link structure of the web subgraph defined by  $S$  to find authority and hub pages in this set
- Displays top authorities in the results

- For a specific query  $Q$ , let the set of documents returned by a standard search engine be called the *root set*  $R$ .
- Initialize  $S$  to  $R$ .
- Add to  $S$  all pages pointed to by any page in  $R$ .
- Add to  $S$  all pages that point to any page in  $R$ .





# Subgraph reduction

- Remove links that serve purely a navigational function
  - For example links within the same domain name
  - E.g., IIT Delhi to IIT Delhi website link
- Remove links that are mentioned in more than  $m$  pages ( $m=4-8$ ).

# Multiple sets of Hubs and Authorities

## Experimental result 1

- For the query “jaguar”, the strongest collections of authoritative sources concerned the Atari Jaguar product, the NFL football team from Jacksonville, and the automobile.

(jaguar\*) Authorities: principal eigenvector

.370 <http://www2.ecst.csuchico.edu/~jschlich/Jaguar/jaguar.html>

.347 <http://www-und.ida.liu.se/~t94patsa/jserver.html>

.292 <http://tangram.informatik.uni-kl.de:8001/~rgehmi/jaguar.html>

.287 <http://www.mcc.ac.uk/dlms/Consoles/jaguar.html>

*Jaguar Page*

(jaguar jaguars) Authorities: 2<sup>nd</sup> non-principal vector, positive end

.255 <http://www.jaguarsnfl.com/>

.137 <http://www.nando.net/SportServer/football/nfl/jax.html>

.133 <http://www.ao.net/~brett/jaguar/index.html>

.110 <http://www.usatoday.com/sports/football/sfn/sfn30.htm>

*Official Jacksonville Jaguars NFL Website*

*Jacksonville Jaguars Home Page*

*Brett's Jaguar Page*

*Jacksonville Jaguars*

(jaguar jaguars) Authorities: 3<sup>rd</sup> non-principal vector, positive end

.227 <http://www.jaguarvehicles.com/>

.227 <http://www.collection.co.uk/>

.211 <http://www.moran.com/sterling/sterling.html>

.211 <http://www.coys.co.uk/>

*Jaguar Cars Global Home Page*

*The Jaguar Collection - Official Web site*

# Multiple sets of Hubs and Authorities

## Experimental result 2

- For the query “randomized algorithms”, none of the strongest collections of hubs and authorities are precisely on the query topic. They include home pages of theoretical computer scientists, compendia of mathematical software and pages on wavelets.

(“randomized algorithms”) Authorities: 1<sup>st</sup> non-principal vector, positive end

.125	<a href="http://theory.lcs.mit.edu/~goemans/">http://theory.lcs.mit.edu/~goemans/</a>	<i>Michel X. Goemans</i>
.122	<a href="http://theory.lcs.mit.edu/~spielman/">http://theory.lcs.mit.edu/~spielman/</a>	<i>Dan Spielman's Homepage</i>
.122	<a href="http://www.nada.kth.se/~johanh/">http://www.nada.kth.se/~johanh/</a>	<i>Johan Hastad</i>
.122	<a href="http://theory.lcs.mit.edu/~rivest/">http://theory.lcs.mit.edu/~rivest/</a>	<i>Ronald L. Rivest : HomePage</i>

(“randomized algorithms”) Authorities 1<sup>st</sup> non-principal vector, negative end

-.00116	<a href="http://lib.stat.cmu.edu/">http://lib.stat.cmu.edu/</a>	<i>StatLib Index</i>
-.00115	<a href="http://www.geo.fni.fi/prog/tela.html">http://www.geo.fni.fi/prog/tela.html</a>	<i>Tela</i>
-.00107	<a href="http://gams.nist.gov/">http://gams.nist.gov/</a>	<i>GAMS : Guide to Available Mathematical Software</i>
-.00107	<a href="http://www.netlib.org">http://www.netlib.org</a>	<i>Netlib</i>

(“randomized algorithms”) Authorities 4<sup>th</sup> non-principal vector, negative end

-.176	<a href="http://www.amara.com/current/wavelet.html">http://www.amara.com/current/wavelet.html</a>	<i>Amara's Wavelet Page</i>
-.172	<a href="http://www-ocean.tamu.edu/~baum/wavelets.html">http://www-ocean.tamu.edu/~baum/wavelets.html</a>	<i>Wavelet sources</i>
-.161	<a href="http://www.mathsoft.com/wavelets.html">http://www.mathsoft.com/wavelets.html</a>	<i>Wavelet Resources</i>
-.143	<a href="http://www.mat.sbg.ac.at/~uhl/wav.html">http://www.mat.sbg.ac.at/~uhl/wav.html</a>	<i>Wavelets</i>

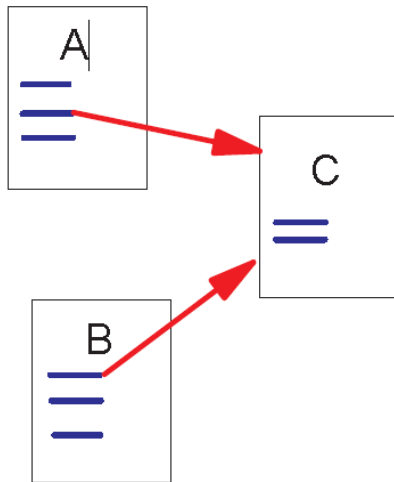
# The Page, Brin's et al. Page Rank Algorithm

# PageRank: History

- PageRank was developed by Larry Page (hence the name *Page*-Rank) and Sergey Brin
- It is first as part of a research project about a new kind of search engine. That project started in 1995 and led to a functional prototype in 1998.
- Shortly after, Page and Brin founded Google.

# Link Structure of the Web

- 150 million web pages → 1.7 billion links



Backlinks and Forward links:

- A and B are C's backlinks
- C is A and B's forward link

Intuitively, a webpage is important if it has a lot of backlinks.

# PageRank: A Simplified Version

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

- $u$ : a web page
- $B_u$ : the set of  $u$ 's backlinks
- $N_v$ : the number of forward links of page  $v$
- $c$ : the normalization factor to make  $\|R\|_{L1} = 1$  ( $\|R\|_{L1} = |R_1 + \dots + R_n|$ )

# Using Matrix Representation

*Definition:* Adjacency matrix of a graph  $G = (V, E)$  is a  $n \times n$  binary matrix  $A$  ( $n = |V|$ ) such that

$$a_{uv} = 1 \text{ if } (u, v) \in E$$

Let  $w = A \mathbf{1}^n$

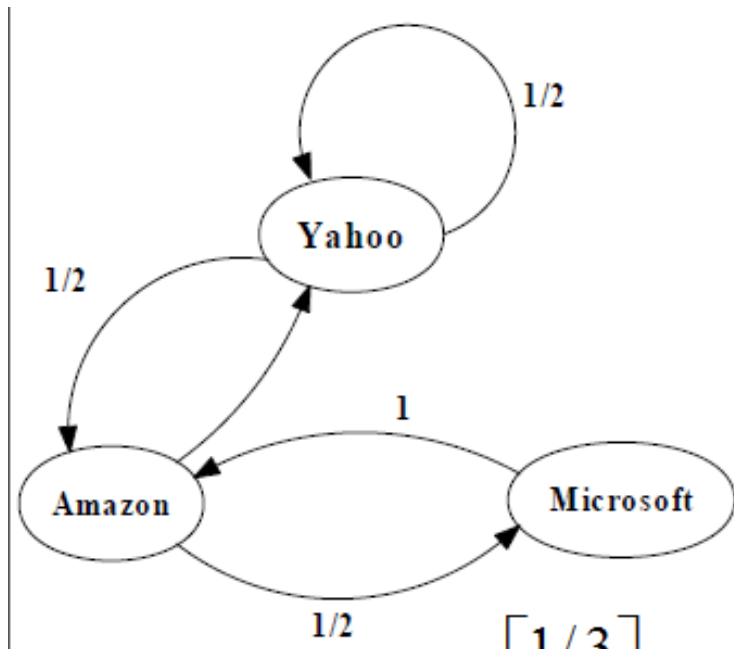
Elements of  $w$  represent the row sum of  $A$

Divide the rows of  $A$  by corresponding element in  $w$  to get the matrix  $B$

Solve for:  $r = cB^T r$



# An example of Simplified PageRank



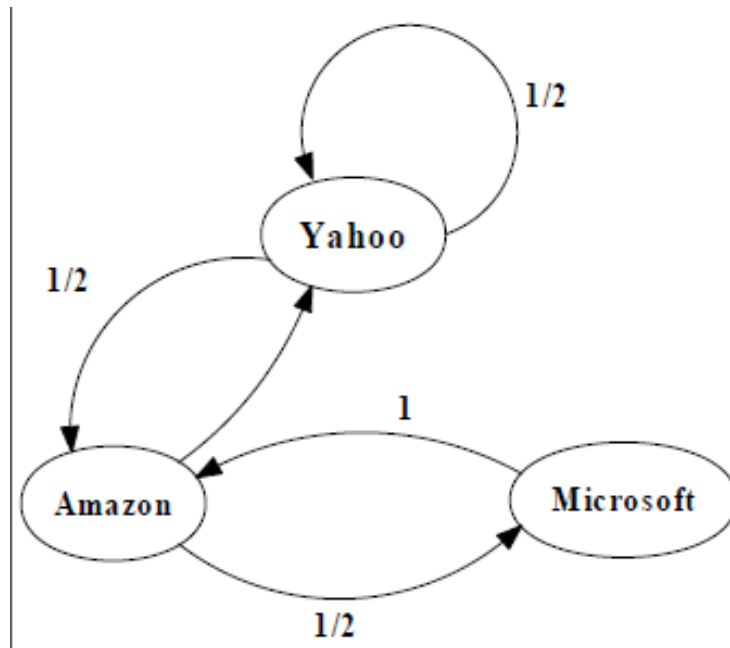
$$B^T = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix}$$

$$\begin{bmatrix} \text{yahoo} \\ \text{Amazon} \\ \text{Microsoft} \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$\begin{bmatrix} 1/3 \\ 1/2 \\ 1/6 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

PageRank Calculation: first iteration

# An example of Simplified PageRank



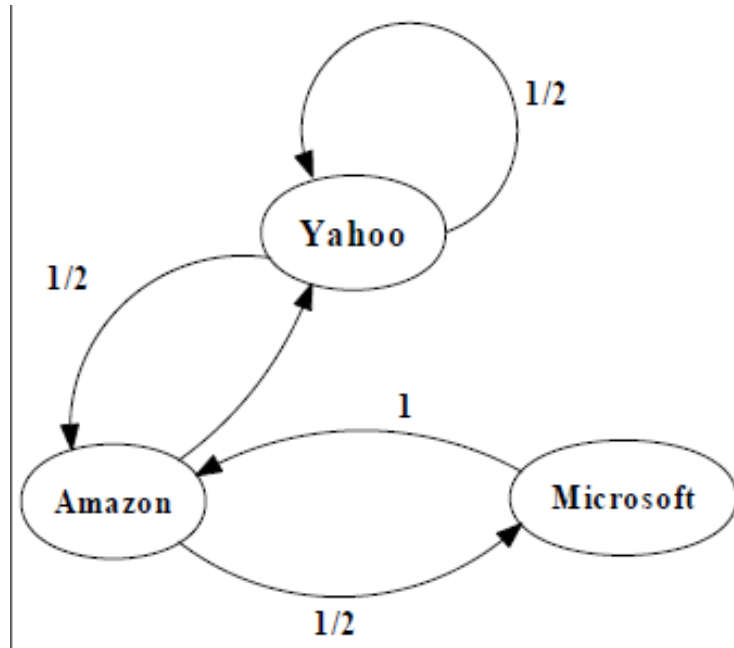
$$\mathbf{B}^T = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix}$$

$$\begin{bmatrix} \text{yahoo} \\ \text{Amazon} \\ \text{Microsoft} \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$\begin{bmatrix} 5/12 \\ 1/3 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/2 \\ 1/6 \end{bmatrix}$$

PageRank Calculation: second iteration

# An example of Simplified PageRank



$$\mathbf{B}^T = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix}$$

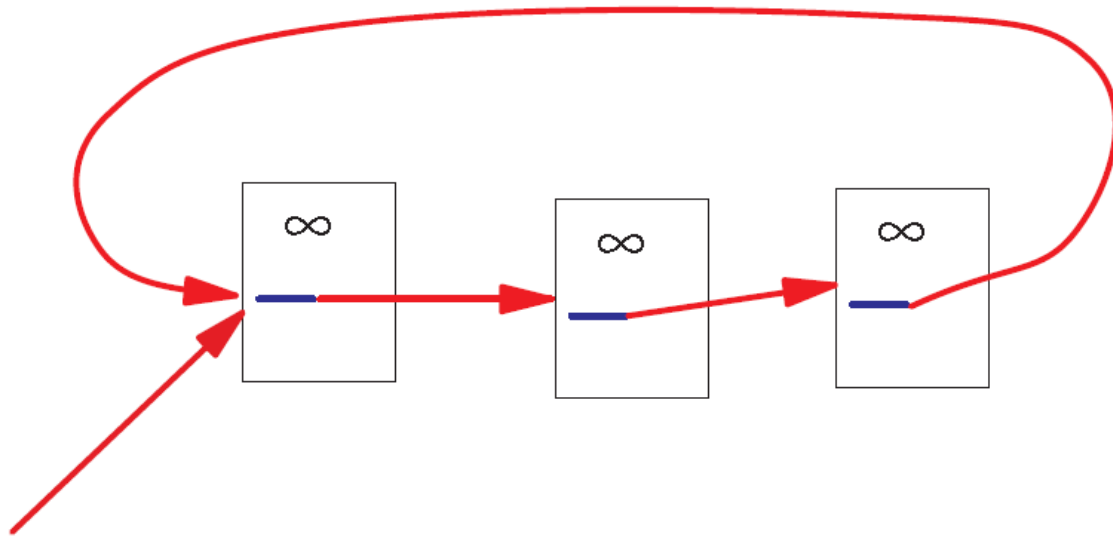
$$\begin{bmatrix} \text{yahoo} \\ \text{Amazon} \\ \text{Microsoft} \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$\begin{bmatrix} 3/8 \\ 11/24 \\ 1/6 \end{bmatrix} \quad \begin{bmatrix} 5/12 \\ 17/48 \\ 11/48 \end{bmatrix} \quad \dots \quad \begin{bmatrix} 2/5 \\ 2/5 \\ 1/5 \end{bmatrix}$$

Convergence after some iterations

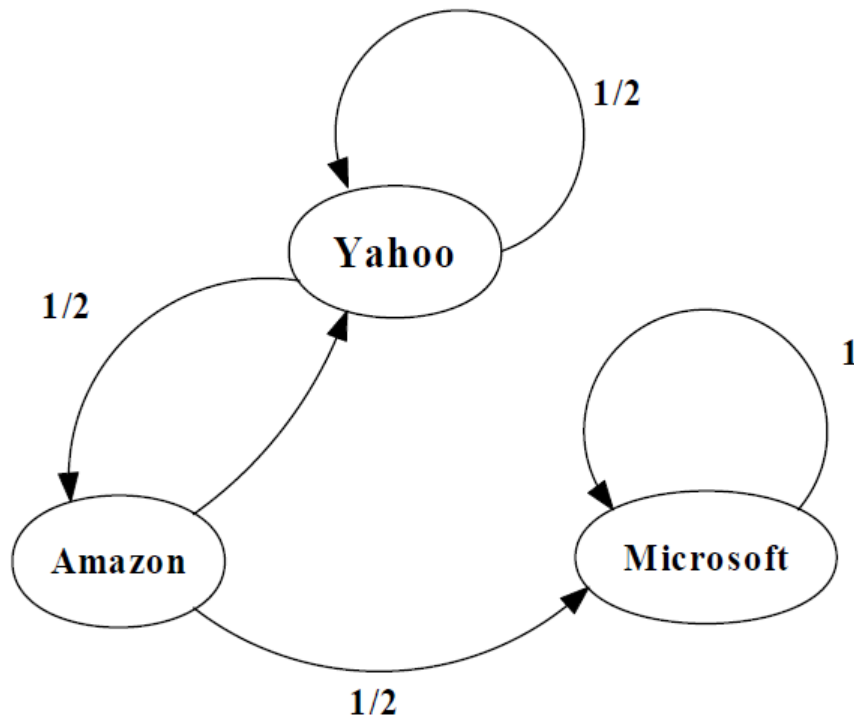
# A Problem with Simplified PageRank

A loop:



During each iteration, the loop accumulates rank but never distributes rank to other pages!

# An example of the Problem

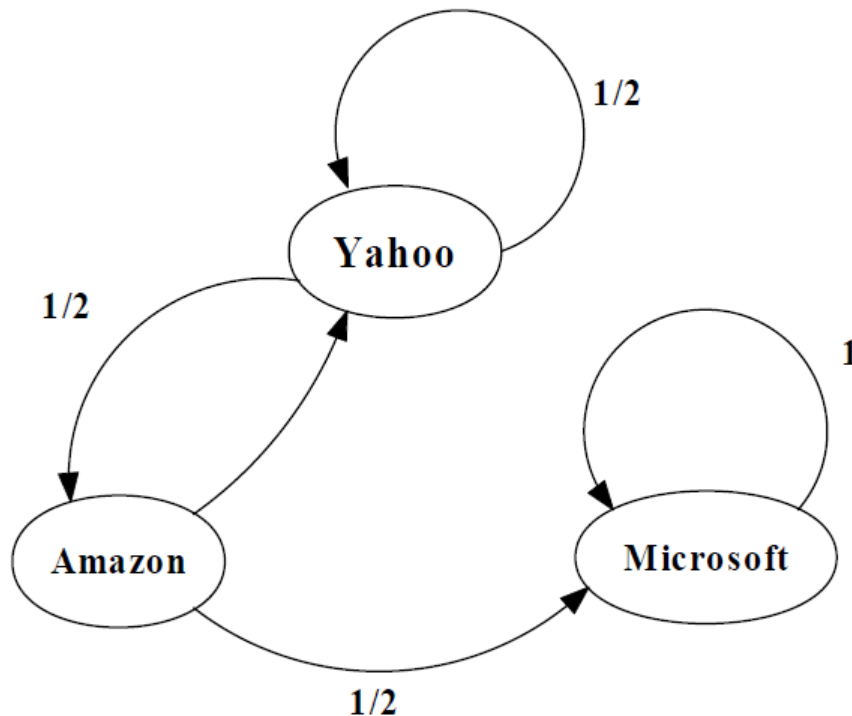


$$\mathbf{B}^T = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}^*$$

$$\begin{bmatrix} \text{yahoo} \\ \text{Amazon} \\ \text{Microsoft} \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$\begin{bmatrix} 1/3 \\ 1/6 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}^*$$

# An example of the Problem

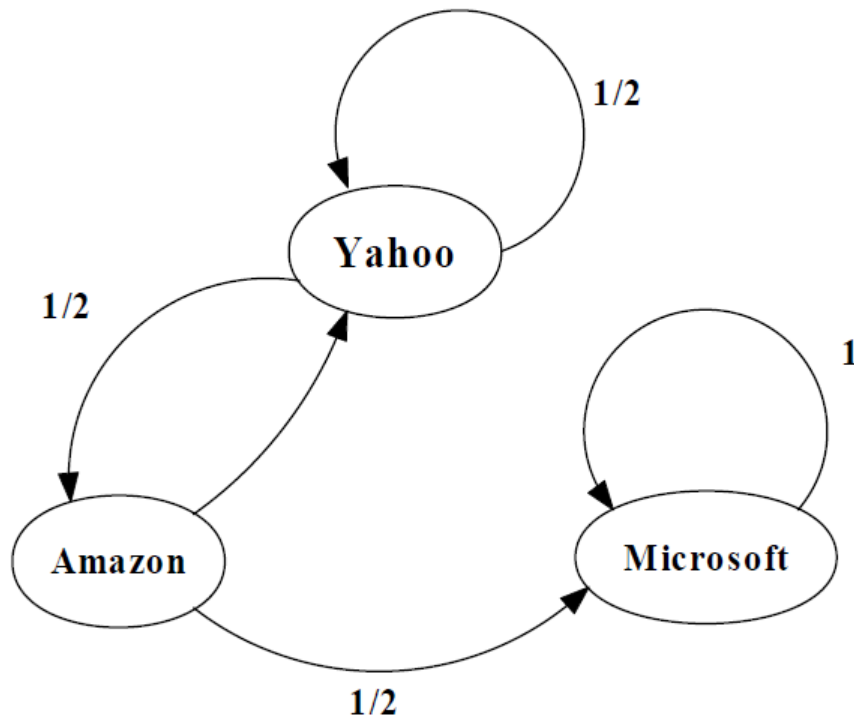


$$\mathbf{B}^T = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}^*$$

$$\begin{bmatrix} \text{yahoo} \\ \text{Amazon} \\ \text{Microsoft} \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$\begin{bmatrix} 1/4 \\ 1/6 \\ 7/12 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/6 \\ 1/2 \end{bmatrix}^*$$

# An example of the Problem



$$B^T = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}^*$$

$$\begin{bmatrix} \text{yahoo} \\ \text{Amazon} \\ \text{Microsoft} \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$\begin{bmatrix} 5/24 \\ 1/8 \\ 2/3 \end{bmatrix} \begin{bmatrix} 1/6 \\ 5/48 \\ 35/48 \end{bmatrix} \dots \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^*$$

# Random Walks in Graphs

- The Random Surfer Model
  - The simplified model: the standing probability distribution of a random walk on the graph of the web. simply keeps clicking successive links at random
- The Modified Model
  - The modified model: the “random surfer” simply keeps clicking successive links at random, but periodically “gets bored” and jumps to a random page based on the distribution of  $E$

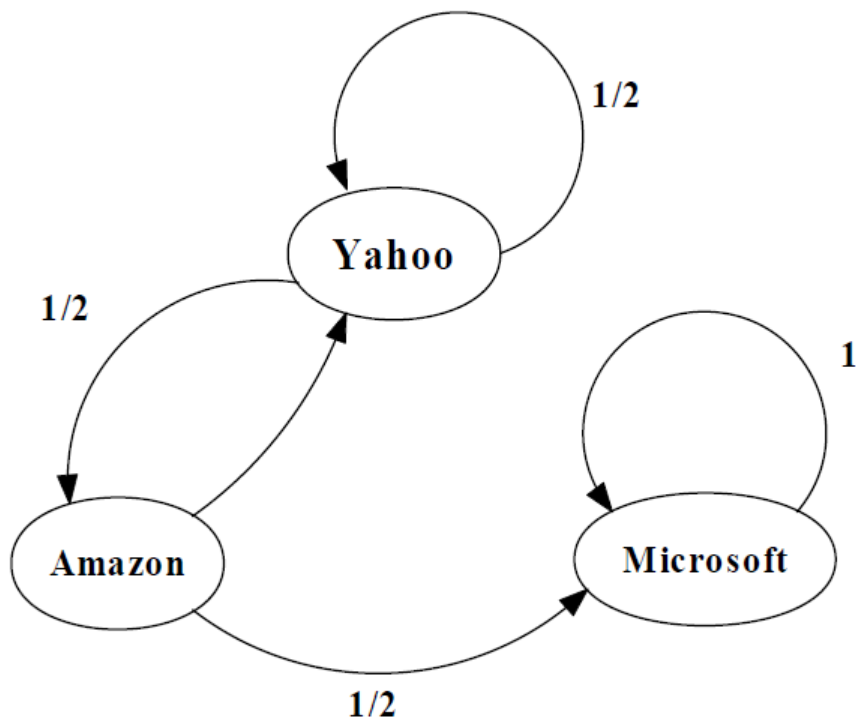


# Modified Version of PageRank

$$R'(u) = c_1 \sum_{v \in B_u} \frac{R'(v)}{N_v} + c_2 E(u)$$

$E(u)$ : a distribution of ranks of web pages that “users” jump to when they “gets bored” after successive links at random.

# An example of Modified PageRank



$$B^T = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}$$

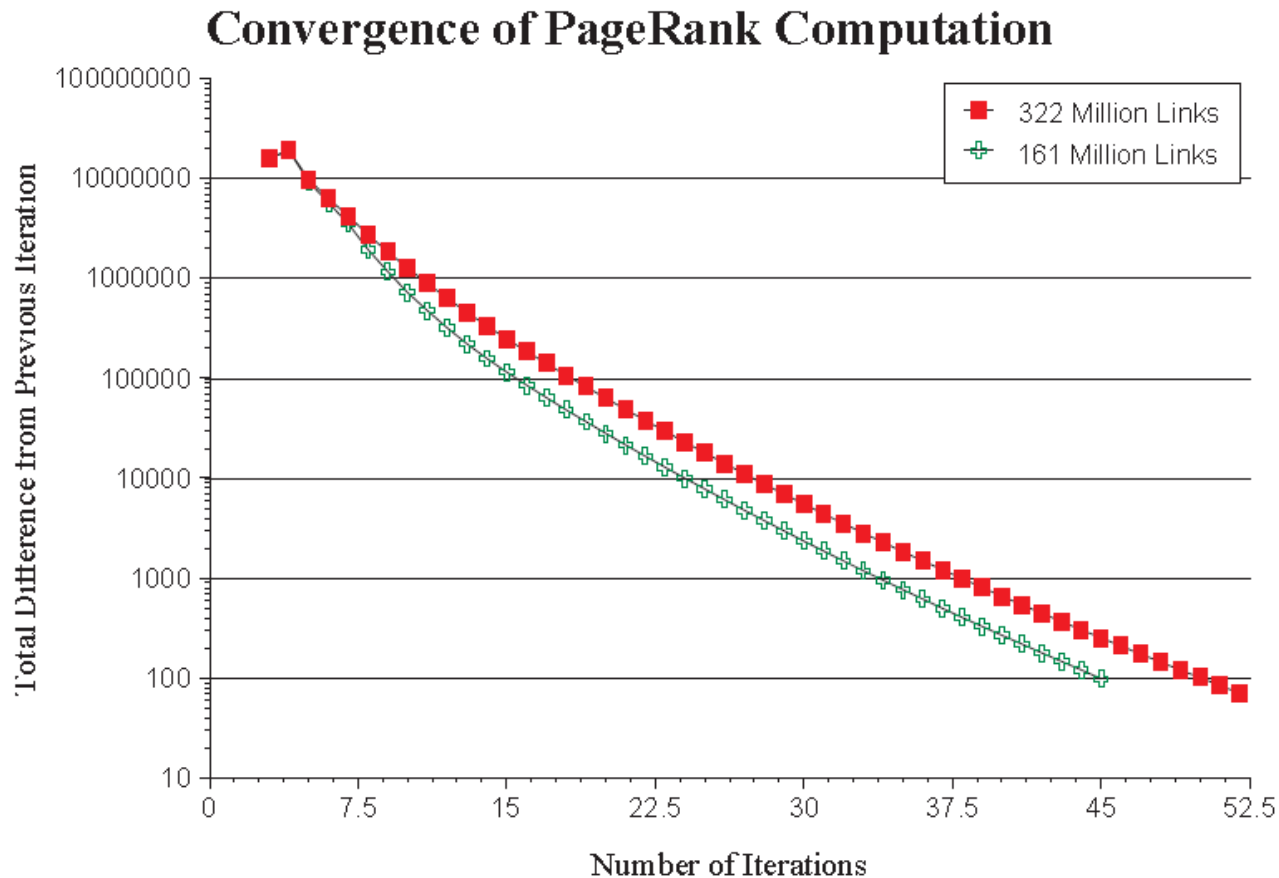
$$\begin{bmatrix} \text{yahoo} \\ \text{Amazon} \\ \text{Microsoft} \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$C_1 = 0.8 \quad C_2 = 0.2$$

$$\begin{bmatrix} 0.333 \\ 0.333 \\ 0.333 \end{bmatrix} \quad \begin{bmatrix} 0.333 \\ 0.200 \\ 0.467 \end{bmatrix} \quad \begin{bmatrix} 0.280 \\ 0.200 \\ 0.520 \end{bmatrix} \quad \begin{bmatrix} 0.259 \\ 0.179 \\ 0.563 \end{bmatrix} \quad \dots \quad \begin{bmatrix} 7/33 \\ 5/33 \\ 21/33 \end{bmatrix}$$

# Convergence Property

- PR (322 Million Links): 52 iterations
- PR (161 Million Links): 45 iterations
- Scaling factor is roughly linear in  $\log n$



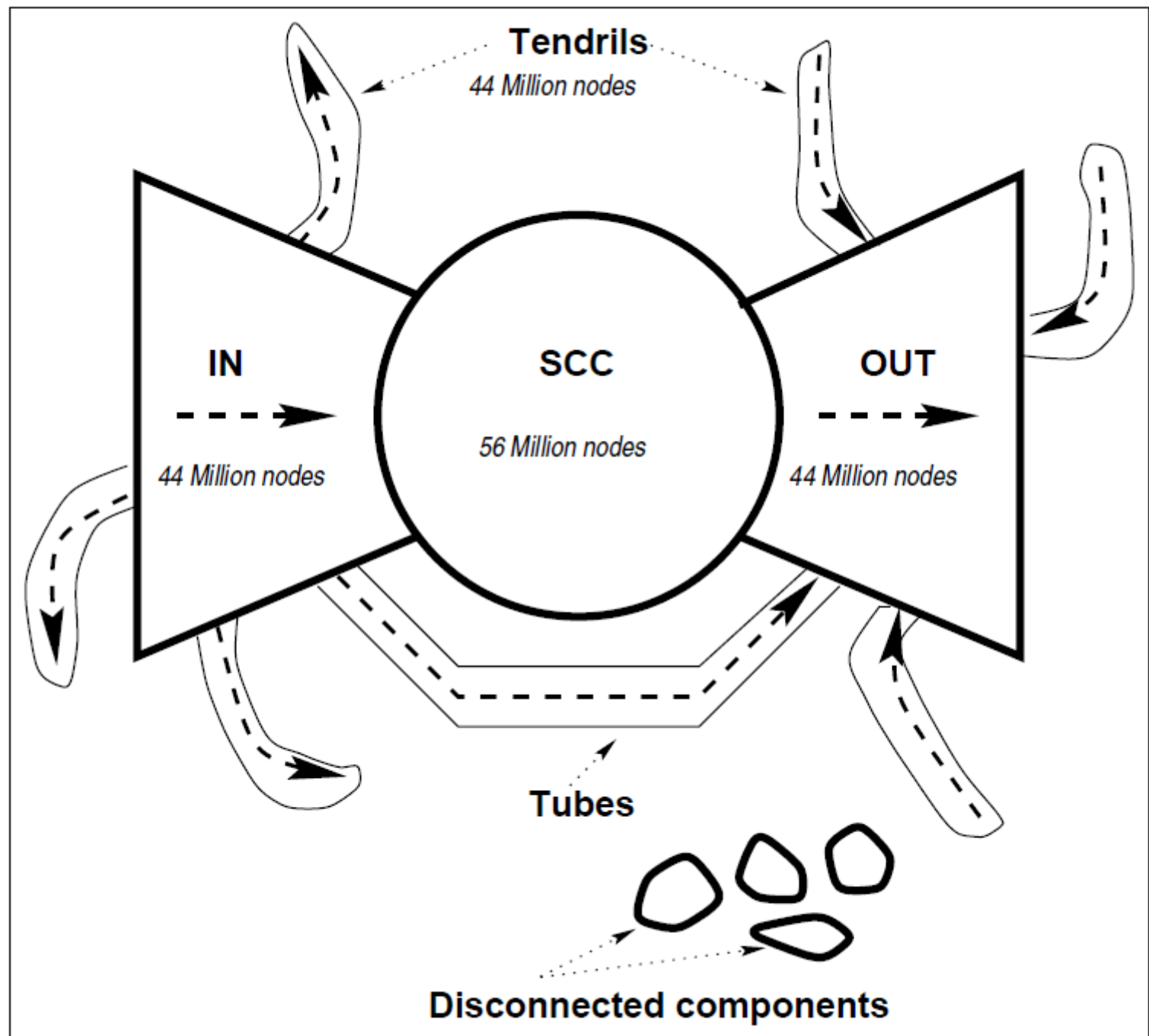
# PageRank v.s. HITS

- PageRank
  - Computed for all web pages stored prior to the query
  - Computes authorities only
  - Fast to compute
- HITS
  - Performed on the subset generated by each query.
  - Computes authorities and hubs
  - Easy to compute, real-time execution is hard.

Which one is more suitable for large scale data set??

# Web as a Graph

How does the Internet www look like?



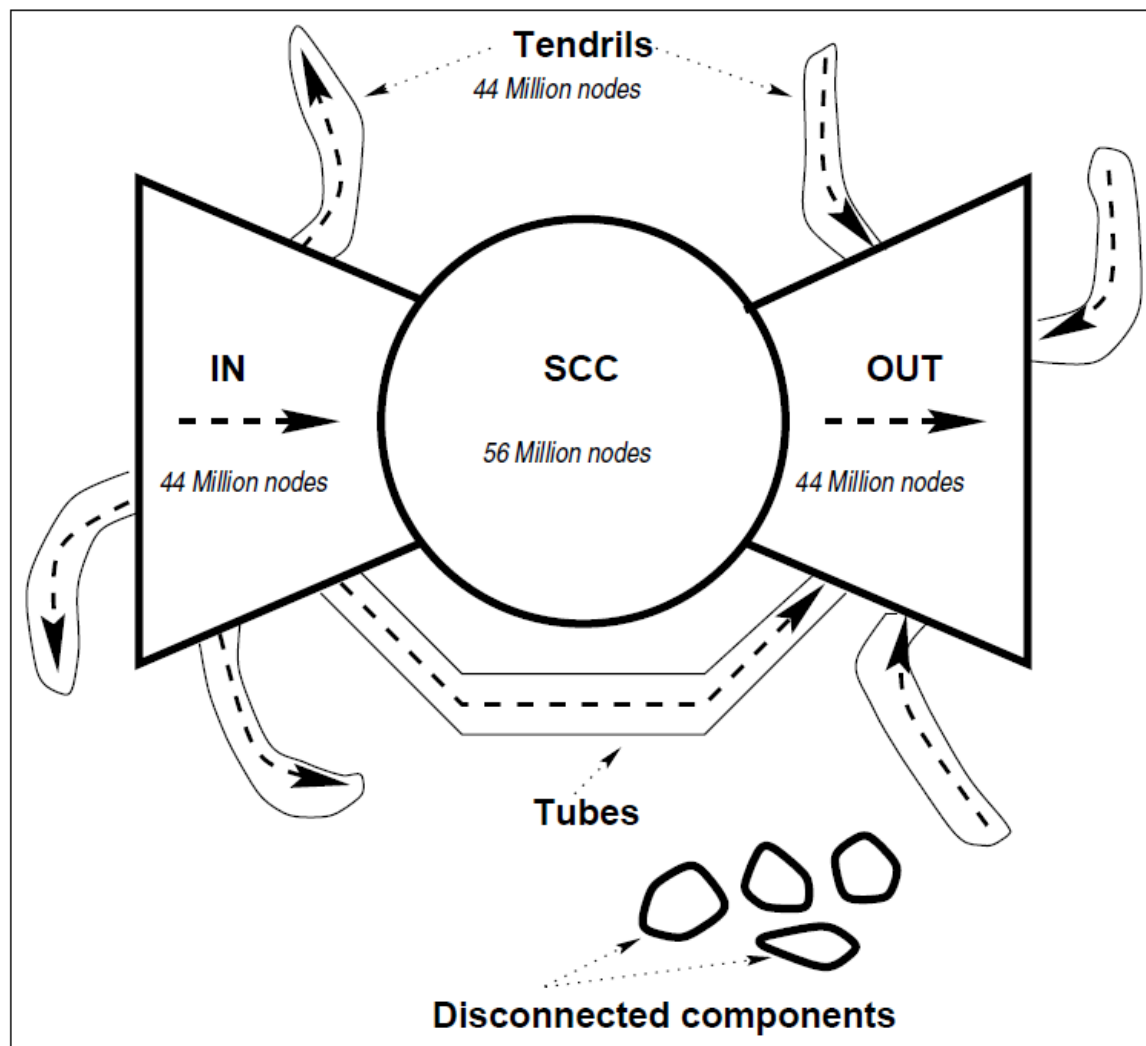


Figure 4: The web as a bowtie. *SCC* is a giant strongly connected component. *IN* consists of pages with paths to *SCC*, but no path from *SCC*. *OUT* consists of pages with paths from *SCC*, but no path to *SCC*. *TENDRILS* consists of pages that cannot surf to *SCC*, and which cannot be reached by surfing from *SCC*.

Thank You