Social and Technological Network Analysis

Lecture 5: Web Search and Random Walks

Dr. Cecilia Mascolo

UNIVERSITY OF CAMBRIDGE

In This Lecture

- We describe the concept of search in a network.
- We describe powerful techniques to enhance these searches.

UNIVERSITY OF CAMBRIDGE

Search

- When searching “Computer Laboratory” on Google the first link is for the department’s page.

- How does Google know this is the best answer?
  - Information retrieval problem: synonyms (jump/leap), polysemy (Leopard), etc
  - Now with the web: diversity in authoring introduces issues of common criteria for ranking documents
  - The web offers abundance of information: whom do we trust as source?
- Still one issue: static content versus real time
  - World trade center query on 11/9/01
  - Twitter helps solving these issues these days

UNIVERSITY OF CAMBRIDGE

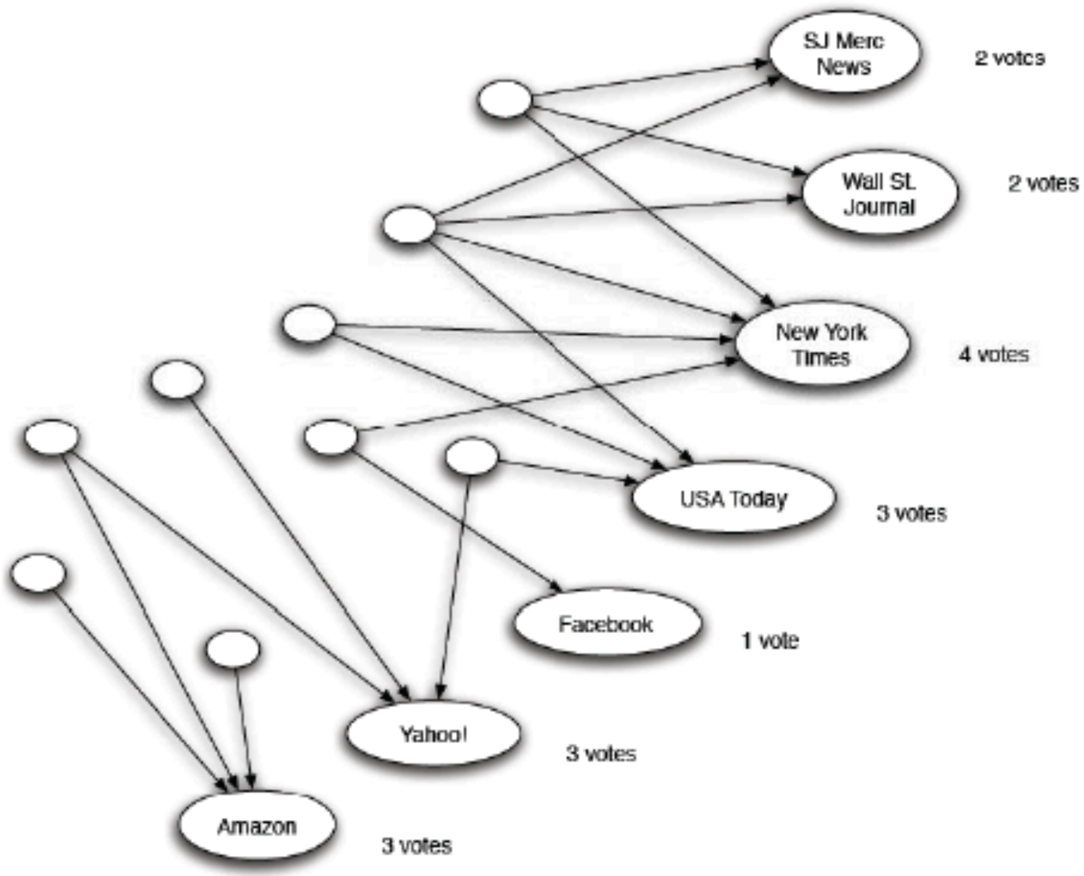
Automate the Search

- We could collect a large sample of pages relevant to “computer laboratory” and collect their votes through their links.
- The pages receiving more in-links are ranked first.
- But if we use **the network structure** more deeply we can improve results.

UNIVERSITY OF CAMBRIDGE

Example: Query "newspaper" Authorities

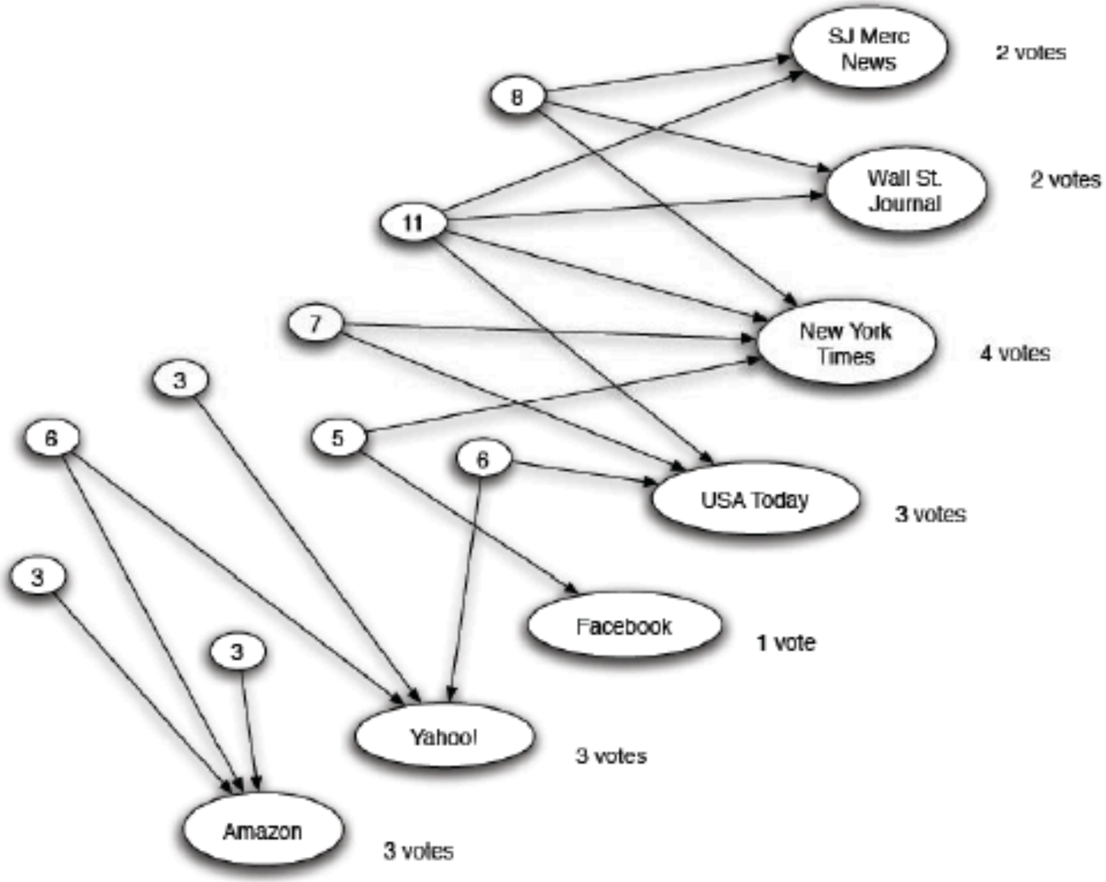
- Links are seen as votes.
- **Authorities** are established: the highly endorsed pages



UNIVERSITY OF CAMBRIDGE

A Refinement: Hubs

- Numbers are reported back on the source page and aggregate.
- Hubs are high value lists

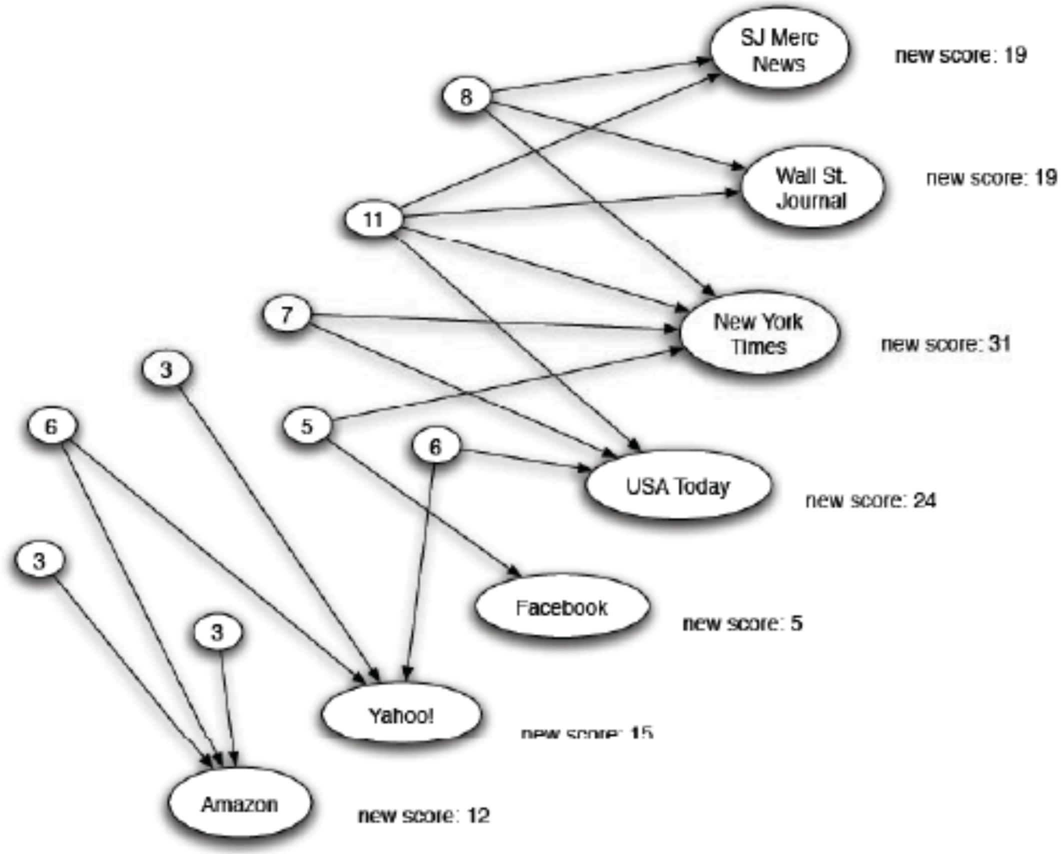


UNIVERSITY OF CAMBRIDGE

Principle of Repeated Improvement

- And we are now reweighting the authorities

- When do we stop?



UNIVERSITY OF CAMBRIDGE

Repeating and Normalizing

- The process can be repeated
- Normalization:
  - Each authority score is divided by the sum of all authority scores
  - Each hub score is divided by the sum of all hub scores

More Formally: does the process converge?

- Each page has an authority a_i and a hub h_i score
- Initially $a_i = h_i = 1$
- At each step

$$a_i = \sum_{j->i} h_j$$

$$h_j = \sum_{j->i} a_i$$

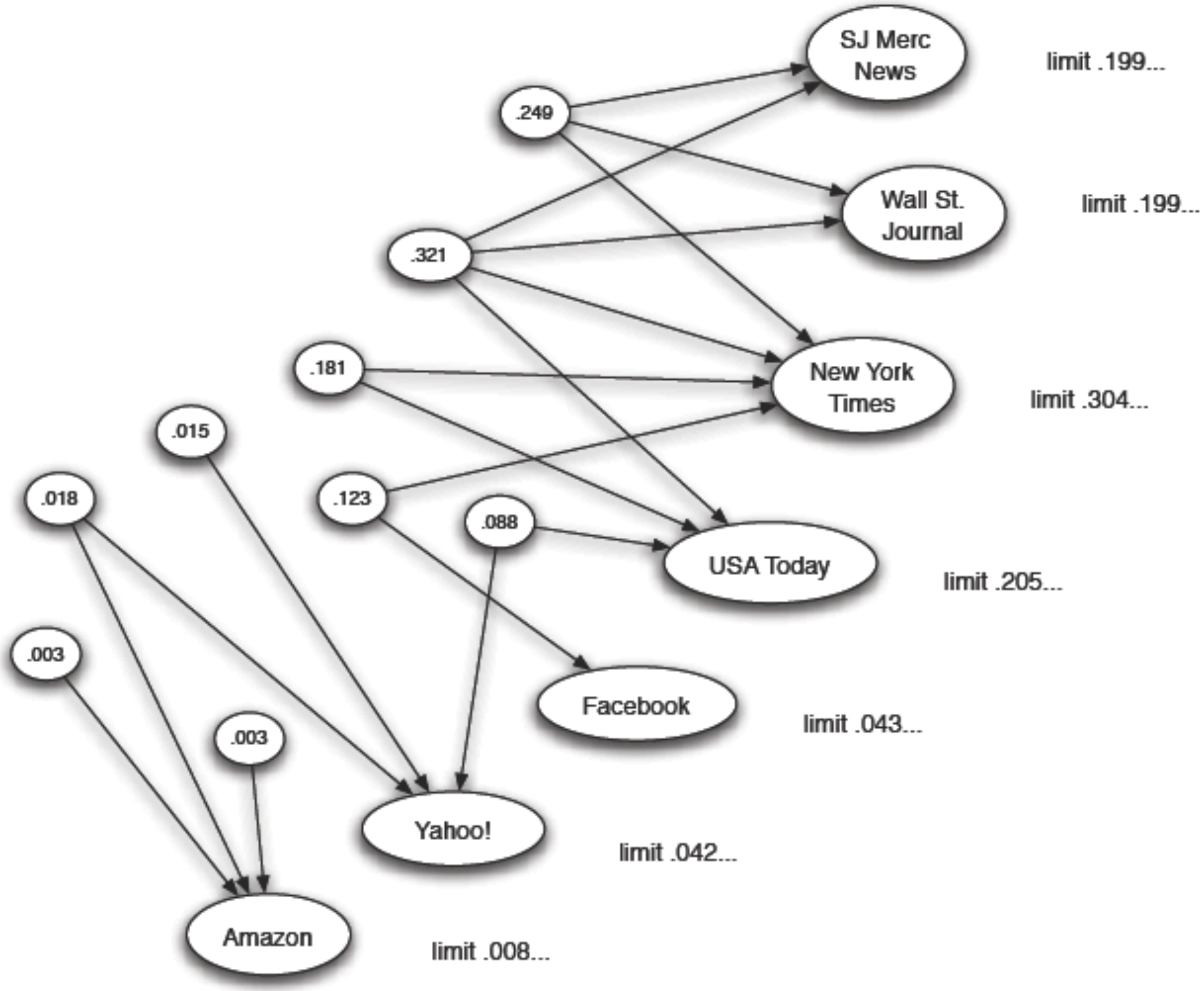
- Normalize

$$\sum a_i = 1$$

$$\sum h_j = 1$$

UNIVERSITY OF CAMBRIDGE

The process converges



UNIVERSITY OF CAMBRIDGE

Let’s prove it

- Graph seen as matrix M_{ij} [where m_{ij}=1 if i->j]
- Vectors a=(a1,a2..an), h=(h1,h2,..hn)

$$h_i = \sum_{i->j} a_j \Leftrightarrow h_i = \sum_j M_{ij} a_j$$

- So $h=Ma$ and a=$M^T h$

UNIVERSITY OF CAMBRIDGE

Algorithm

- Initially $a=h=1^n$
- Step
  - $h=Ma$, $a=M^Th$
  - Normalize
- Then $a=M^T(Ma)=(M^TM)a$
- And $h=M(M^Th)=(MM^T)h$

- In 2k steps
  $a=(M^TM)^k a$ and $h=(MM^T)^k h$

UNIVERSITY OF CAMBRIDGE

Eigenvalues and eigenvectors

- **Definition:**
- *Let $Ax=\lambda x$ for scalar λ, vector x and matrix A*
- *Then x is an eigenvector and λ is its eigenvalue*

- If A is symmetric ($A^T=A$) then A has n orthogonal unit eigenvectors $w_1, \ldots w_n$ that form a basis (ie linear independent) with eigenvalues $\lambda_1,\ldots\lambda_n$ ($|\lambda_i| \leq |\lambda_{i+1}|$)
  - Tip: M^TM and MM^T are symmetric

UNIVERSITY OF CAMBRIDGE

So…

- Given the linear independence of the w_i we can write $x=\Sigma p_i w_i$
- And x has coordinates [p1.. pn]
- If λ1,…λn (|λi|≤|λi+1|) then

- $A^k x=(\lambda_1^k p_1 w_1 + \lambda_2^k p_2 w_{2,\dots+} \lambda_n^k p_n w_n)=\Sigma_. \lambda_i^k p_i w_i$

- Let's divide by λ_1^k
  $A^k x/\lambda_1^k=\lambda_1^k p_1 w_{1/}\lambda_1^k + \lambda_2^k p_2 w_{2/}\lambda_1^k{}_{\dots+.} \lambda_n^k p_n w_{n/}\lambda_1^k$

- When k->∞, $A^k x/\lambda_1^k \to p_1 w_1$

UNIVERSITY OF CAMBRIDGE

Convergence

- $(M^T M)^k a$ and $(M M^T)^k h$
- So for k->∞ these sequences converge

PageRank

- We have seen hubs and authorities
  - Hubs can "collect" links to important authorities who do not point to each others
  - There are other models: better for the web, where one prominent can endorse another.
- The **PageRank** model is based on transferrable importance.

UNIVERSITY OF CAMBRIDGE

PageRank Concepts

- Pages pass endorsements on outgoing links as fractions which depend on out-degree
- Initial PageRank value of each node in a network of n nodes: 1/n.
- Choose a number of steps k.
- **[Basic] Update rule**: each page divides its pagerank equally over the outgoing links and passes an equal share to the pointed pages. Each page's new rank is the sum of received pageranks.

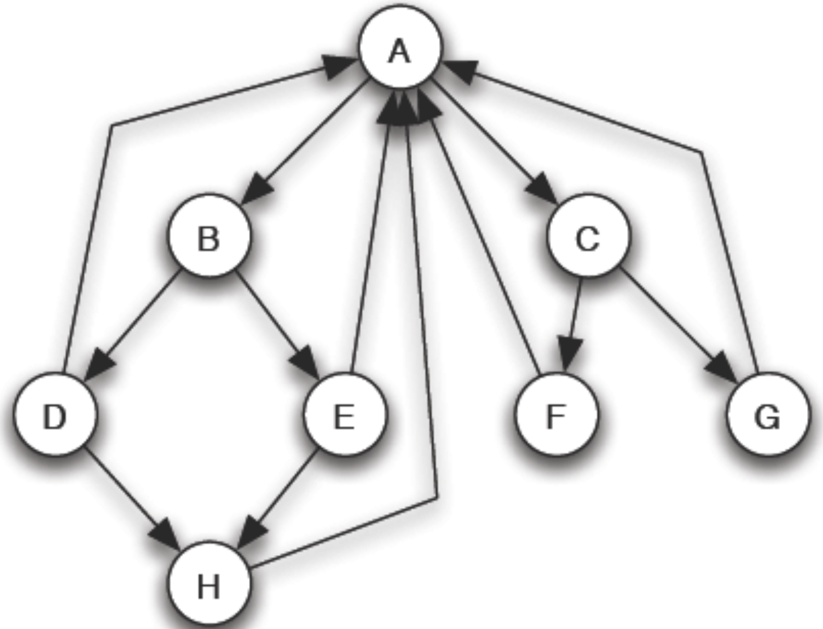
UNIVERSITY OF CAMBRIDGE

Example

- All pages start with PageRank= 1/8

Step	A	B	C	D	E	F	G	H
1	1/2	1/16	1/16	1/16	1/16	1/16	1/16	1/8
2	3/16	1/4	1/4	1/32	1/32	1/32	1/32	1/16

A becomes important and
B,C benefit too at step 2



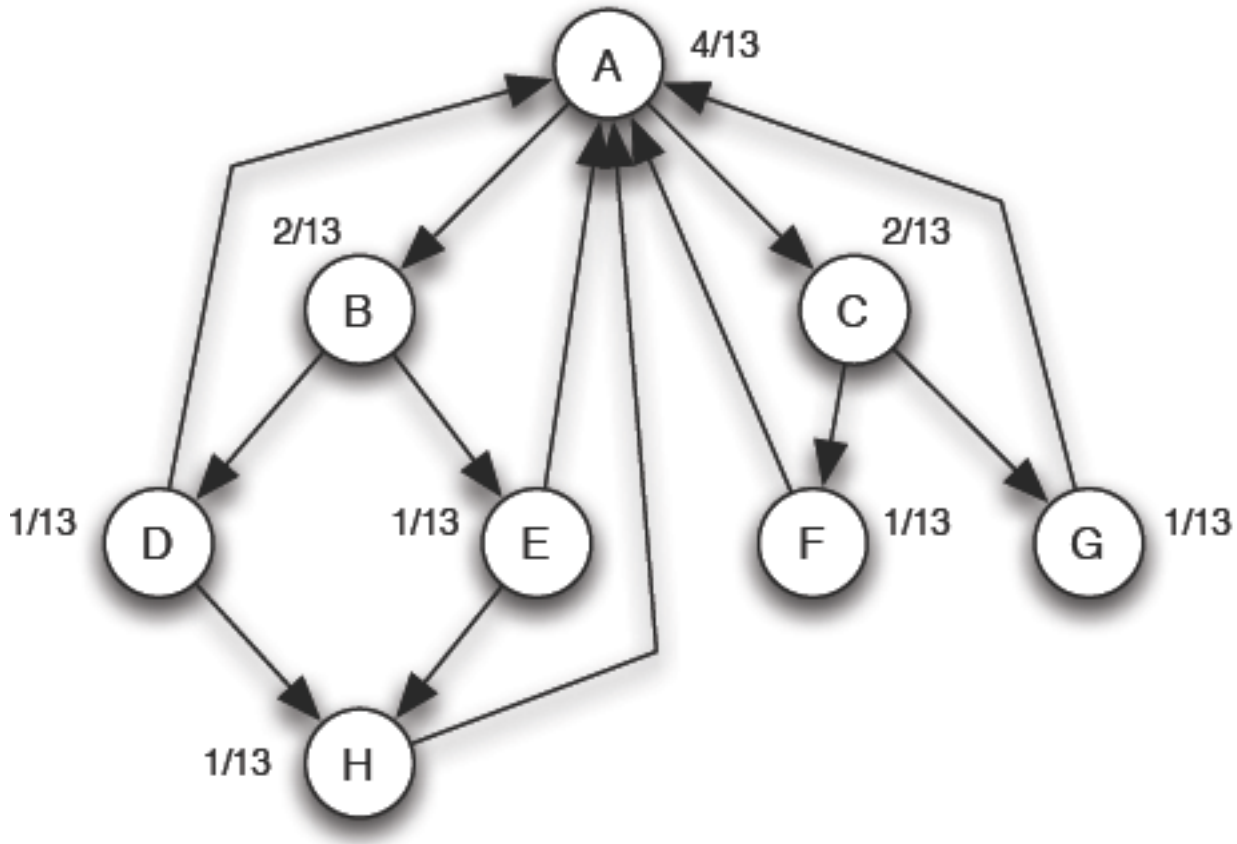
UNIVERSITY OF CAMBRIDGE

Convergence

- Except for some special cases, PageRank values of all nodes converge to limiting values when the number of steps goes to infinity.
- The convergence case is one where the PageRank of each page does not change anymore, i.e., they regenerate themselves.

UNIVERSITY OF CAMBRIDGE

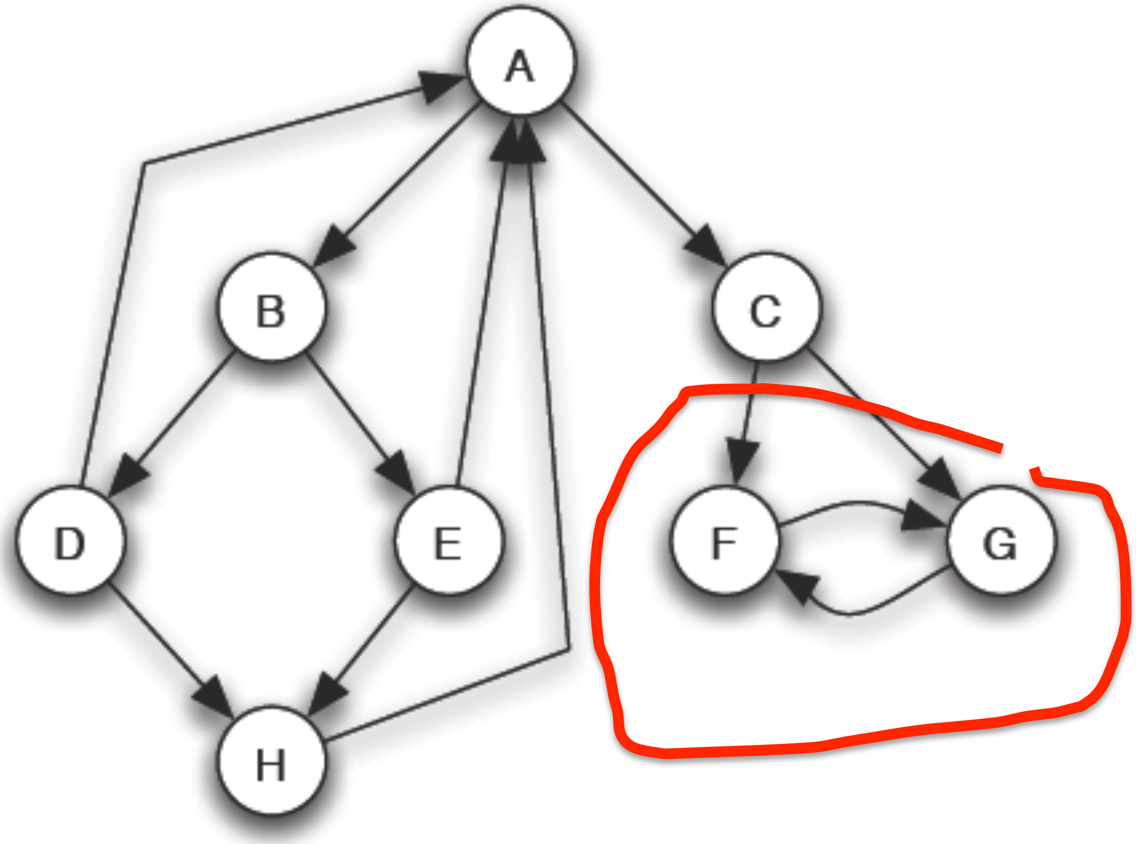
Example of Equilibrium



UNIVERSITY OF CAMBRIDGE

Problems with the basic PageRank
Dead ends

- F,G converge to ½ and all the other nodes to 0



UNIVERSITY OF CAMBRIDGE

Solution: The REAL PageRank

- **[Scaled] Update Rule:**
  - Apply basic update rule. Then, scale down all values by scaling factor s [chosen between 0 and 1].
  - **[Total network PageRank value changes from 1 to s]**
  - Divide 1-s residual units of PageRank equally over all nodes: (1-s)/n each.

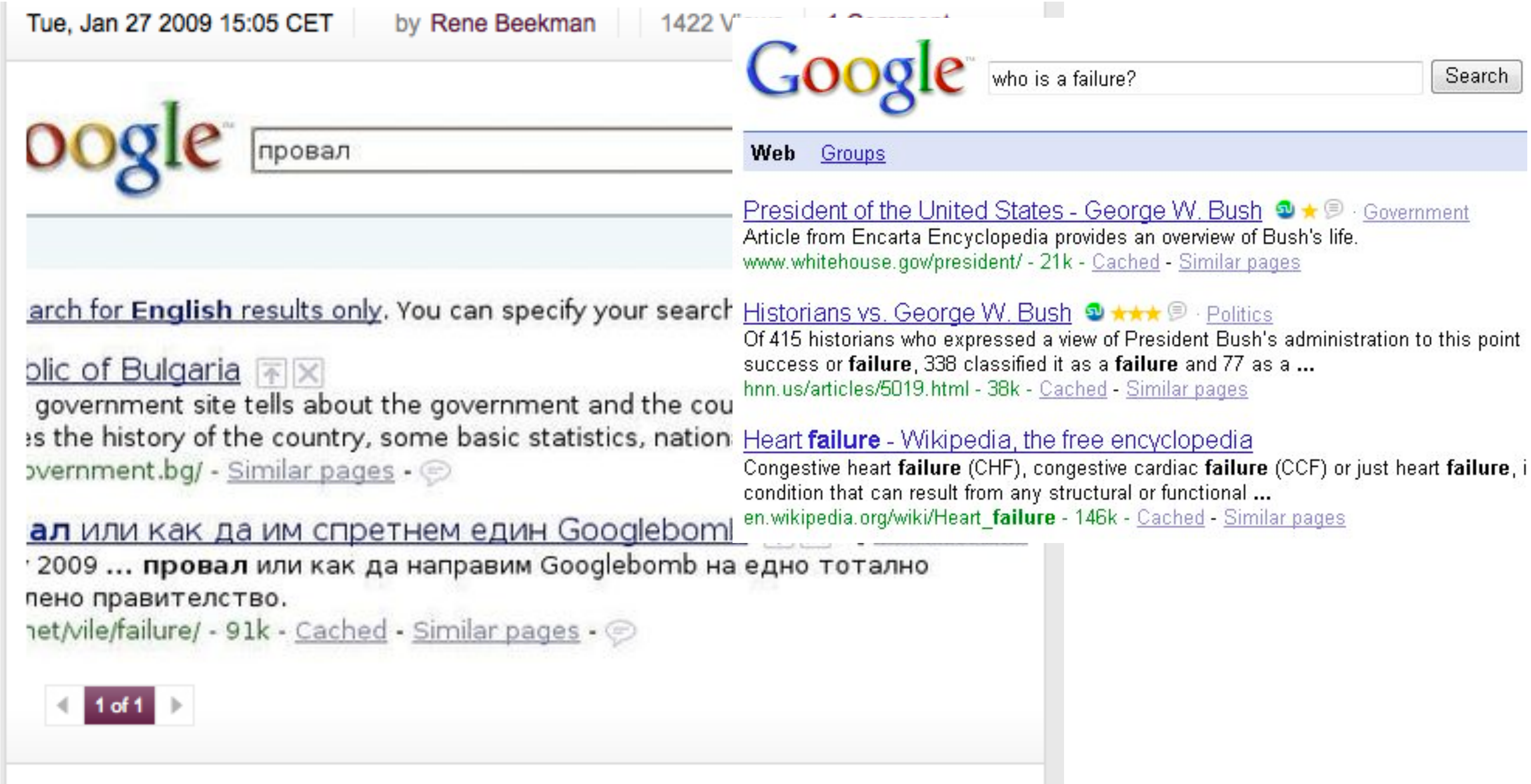
- It can be proven that values converge again.
- Scaling factor usually chosen between 0.8 and 0.9

UNIVERSITY OF CAMBRIDGE

Search Ranking is very important to business

- A change in results in the search pages might mean loss of business
  - I.e., not appearing on first page.
- Ranking algorithms are kept very secret and changed continuously.

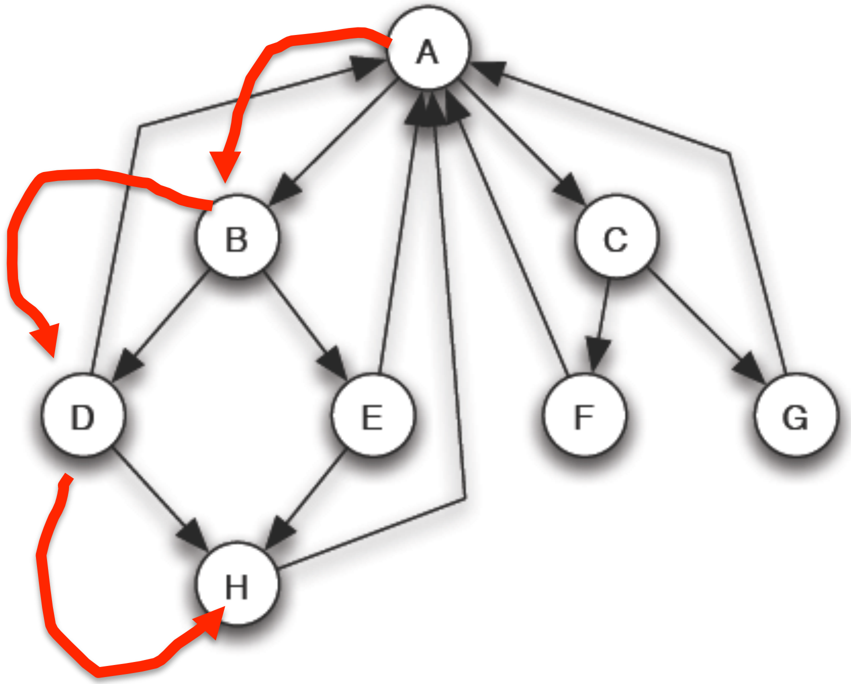
Examples of Google Bombs



UNIVERSITY OF CAMBRIDGE

Random Walks

- Starting from a node, follow one outgoing link with random probability



UNIVERSITY OF CAMBRIDGE

PageRank as Random Walk

- The probability of being at a page X after k steps of a random walk is precisely the PageRank of X after k applications of the Basic PageRank Update Rule
- Scaled Update Rule equivalent: follow a random outgoing link with probability s while with probability 1-s jump to a random node in the network.

References

- Chapter 14