Web Search

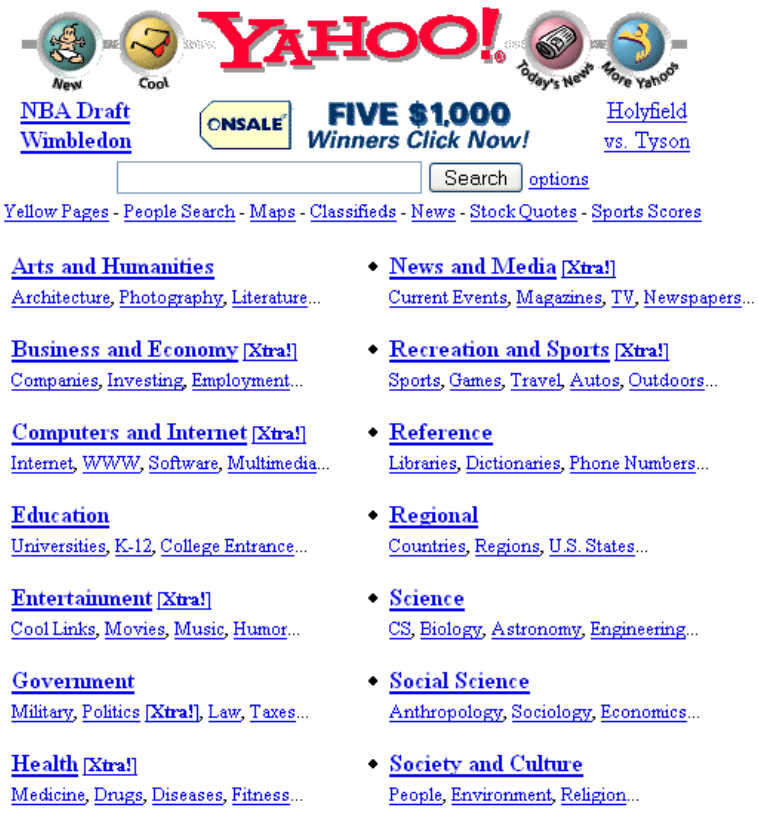
Pagerank and HITS Algorithms

- Imdadullah Khan

Web Search and Information Retrieval

Web Search and Information Retrieval

Initial Solution: Information organized in a directory structure for browsing

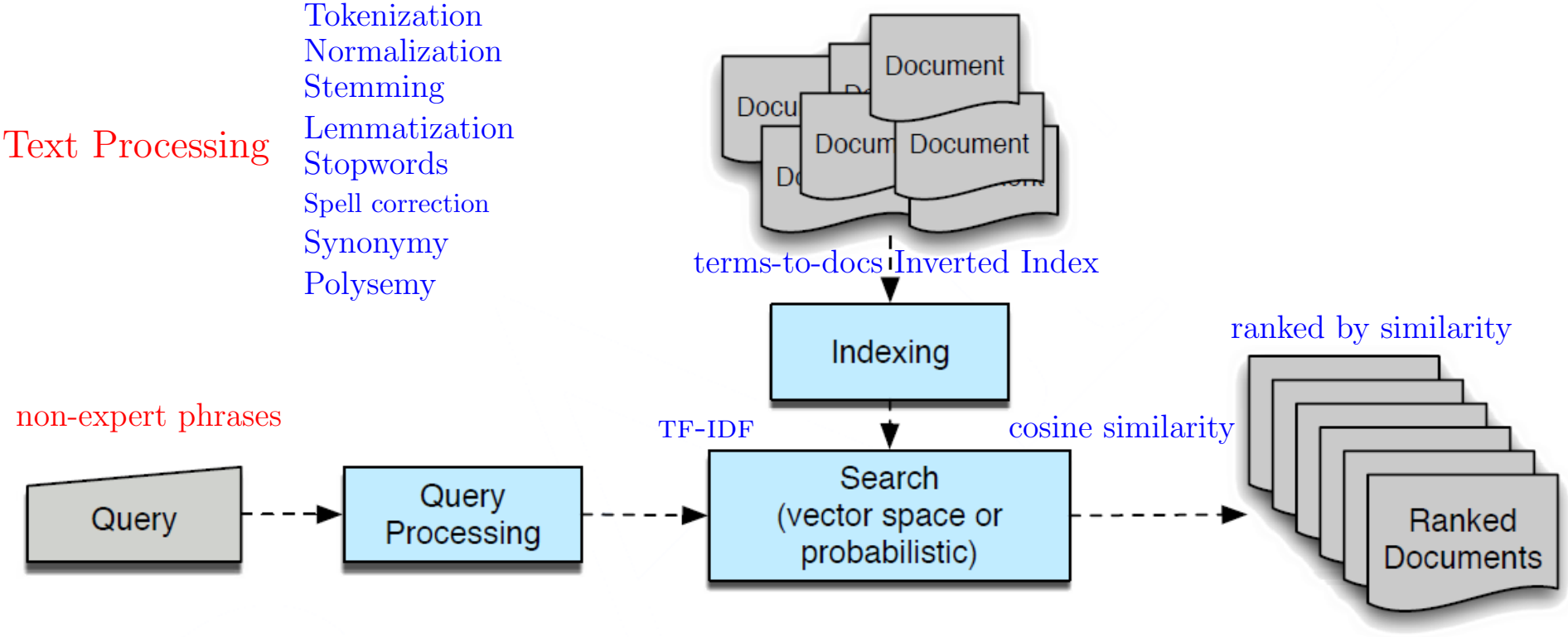


- Human edited - Yahoo listed webpages as standard and paid versions
- initially called Jerry and David's Guide to the World Wide Web
- DMOZ (directory.mozilla.org) maintained by volunteers (Open Directory Project)

Web Search and Information Retrieval

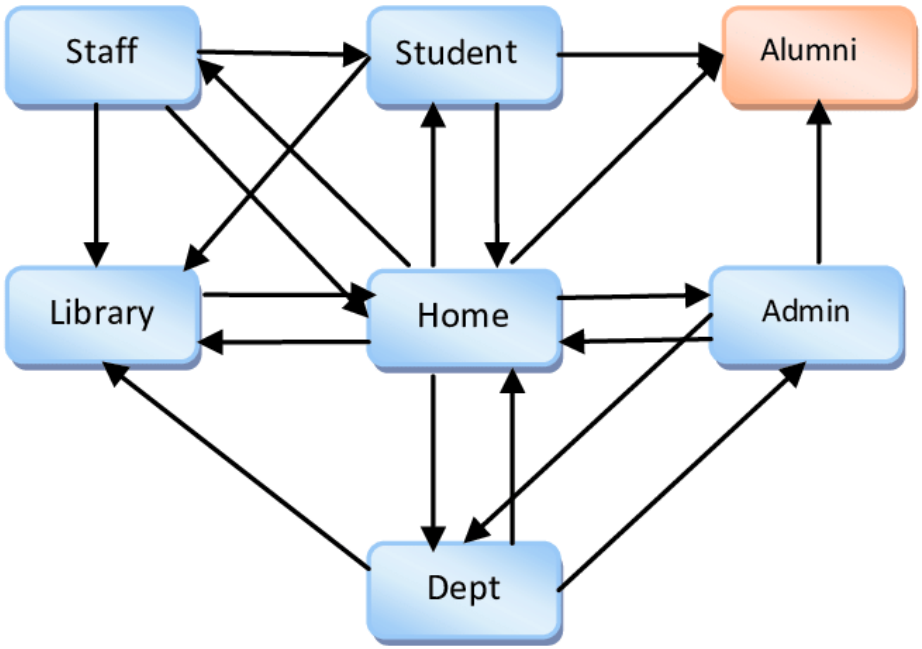
Text Processing

Tokenization
Normalization
Stemming
Lemmatization
Stopwords
Spell correction
Synonymy
Polysemy

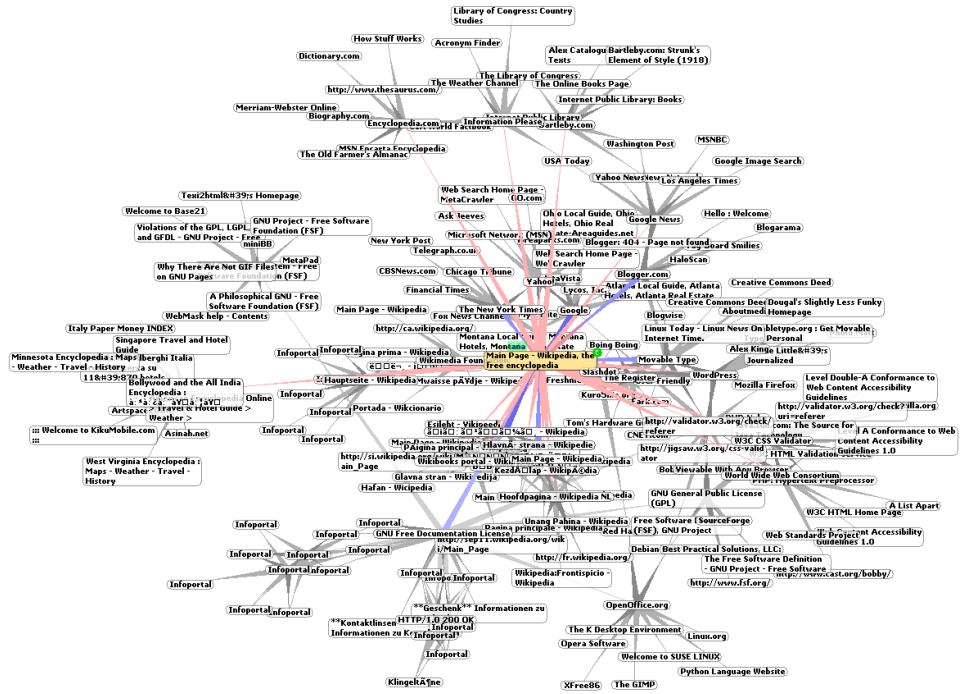


The Web Graph

- A directed graph with webpages as vertices $v_1, v_2, \ldots$
- Page i has a hyperlink to page j, *implies* $(v_i, v_j) \in E$



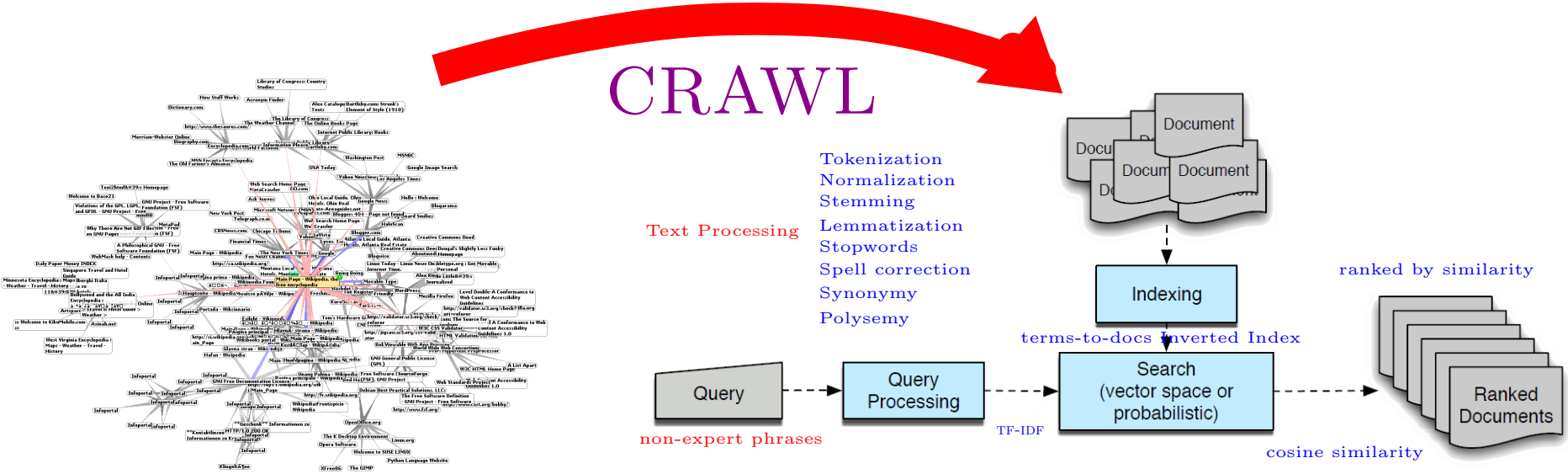
A sample Web Graph W of a University. source: A Singh (2013)



Web Search and Information Retrieval

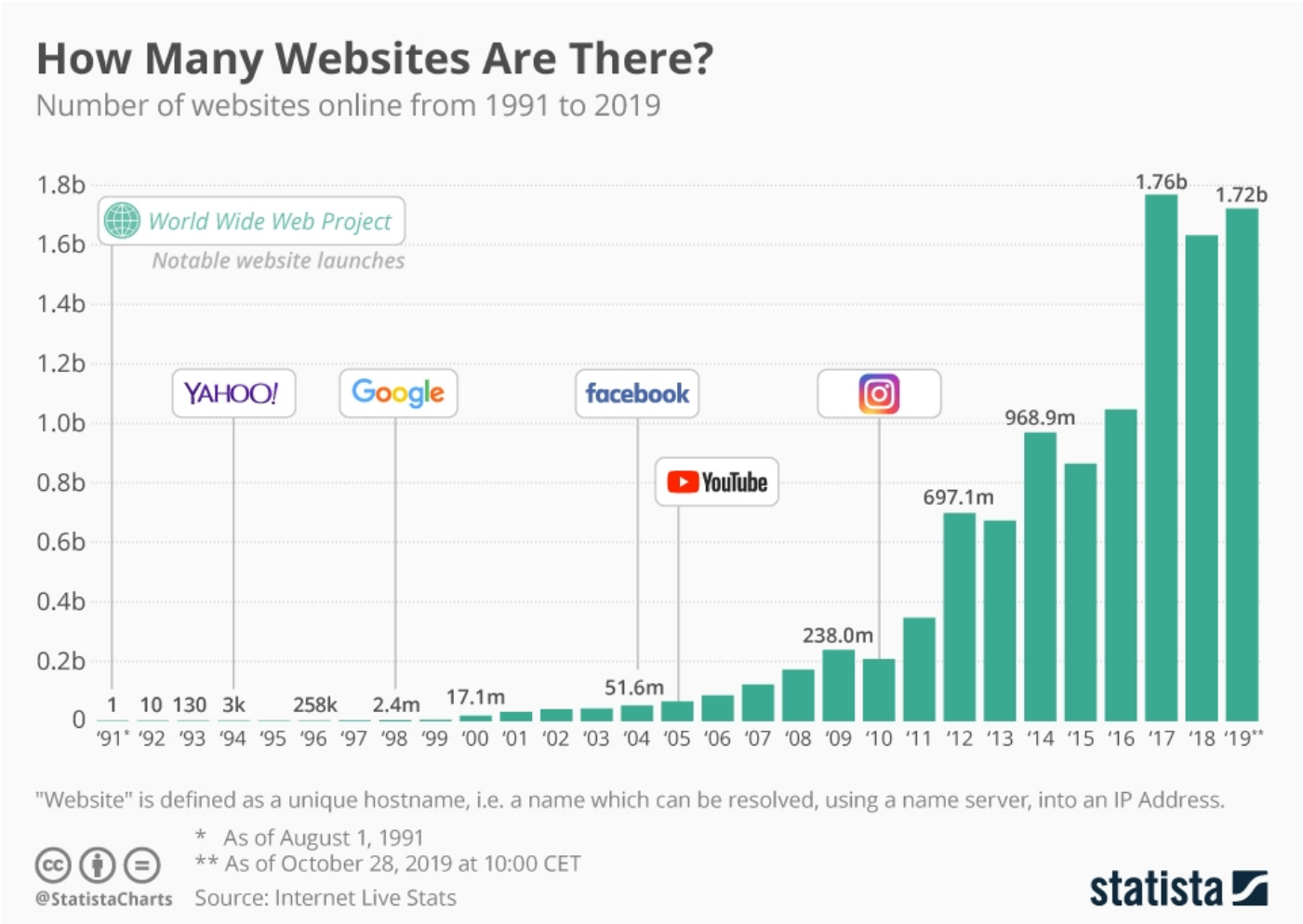
The webgraph is CRAWLED to get the collection of webpages

- Web crawler (aka spiderbot): Internet bot to systematically traverse the web graph (think BFS/DFS starting from a few seeds)
- Collects webspages for (web) indexing (aka web spidering)
- Crawling is also used for web archiving



Web Information Retrieval: Challenges

- 1.83 billion websites, $\sim$ 1.58 billion inactive ▷ internetlivestats, 2021
- Google indexed $\sim$ 55.2 billion webpages in Jan 2021, ▷ worldwidewebsize
- Google Search index is well over 100,000,000 gigabytes

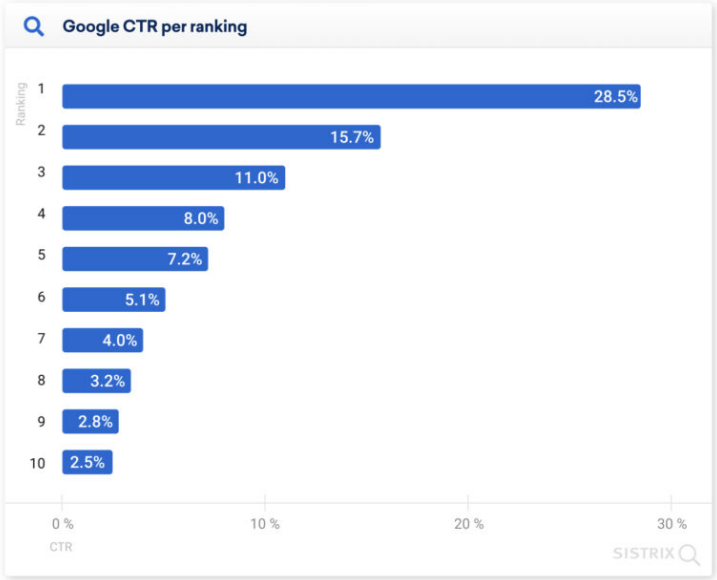


Web Information Retrieval: Challenges

- Retrieve web pages from inverted index
- Rank by cosine similarity b/w TF-IDF vectors of page and query phrase

Easy to manipulate to attract traffic to a web address with financial incentives

- Daily 3.5 billion Google searches – $\sim$ 10% growth p.a ▷ internetlivestats, 2019
- 35% of product searches start on Google ▷ eMarketer
- 46% of product searches begin with Google ▷ Jumpshot, 2018
- 90% of a survey respondents: likely to click on the first set of results ▷ searchengineland, 2018



Web Information Retrieval: Challenges

Financial incentive for traffic to a webpage $\Longrightarrow$ tricks to increase relevance of webpages to multiple keywords

- Termed as *search engine optimization* or *search engine spamming*
  - Depending on which side it is coming from
- Two broad categories of tricks (spamdexing) are
  - Content Spamming:
    - Content manipulation to attract traffic
  - Link Spamming:
    - Graph Structure (hyperlinks) manipulation

Web Information Retrieval: Content Spamming

The goal here is to increase TF-IDF scores of (many) keywords

- Keyword Stuffing: placing many keywords on webpage
  - Search engines truncate large pages, avoided with multiple pages
- Invisible Text: Background-colored text or hidden within html codes
- Doorway Pages: Pages with targeted keywords redirecting traffic to another page aka Cloaking
- Scrapper Sites: Pages using contents of other pages (e.g. top results against a keyword)
- Article spinning or translation: to avoid penalty for duplicate contents - contain rephrased or machine translated articles
- Deceiving Page Titles: Page titles irrelevant to content (title and header terms carry higher scores)

Trustworthy Webpages

- Assign a trustworthiness (popularity/reliability) score to each page
  - Use it in addition to relevance/similarity of page with query keyword
- Page score is based on its "location" in the webgraph ▷ Link Analysis

Node Centrality in Graphs

Node centrality (called prestige in digraphs e.g. Twitter or Web)

- Degree centrality: how many neighbors a node has

$$C_d(v) := deg(v)$$

- Closeness centrality: how "close" a node is to other nodes

$$C_{close}(v) := \frac{1}{\sum_{u \neq v \in V} d_G(v,u)}$$

- Betweenness centrality: how often a nodeis on the shortest paths

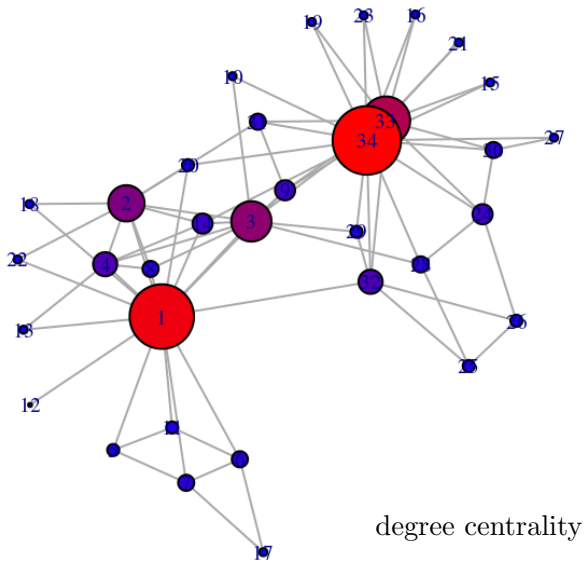
$$C_{bw}(v) := \sum_{s,t \neq v \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

$\sigma_{st}(v)$: number of shortest paths between s and t through v
σ_{st}: number of shortest paths between s and t

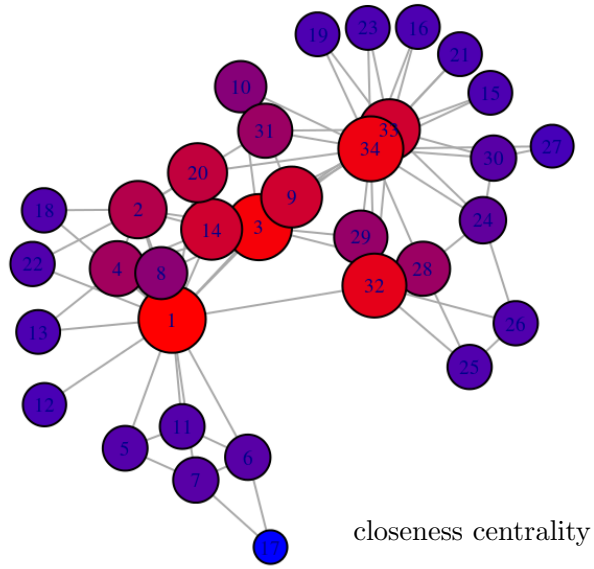
- Eigenvector centrality: Value of eigenvector at corresponding coordinate

$\mathbf{x}[i]$, $\mathbf{x}$: eigen vector correspond. to leading eigenvalue

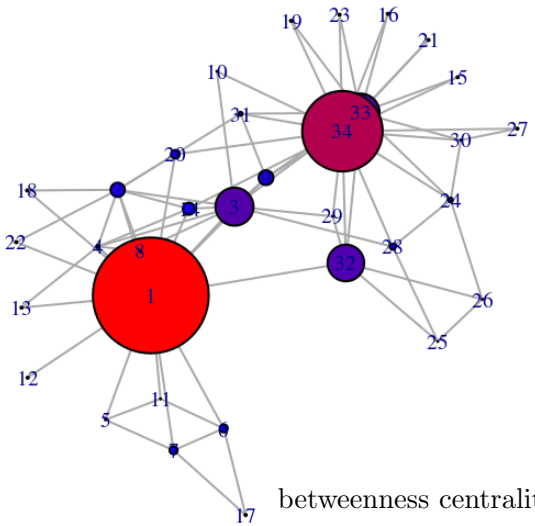
Node Centrality in Graphs



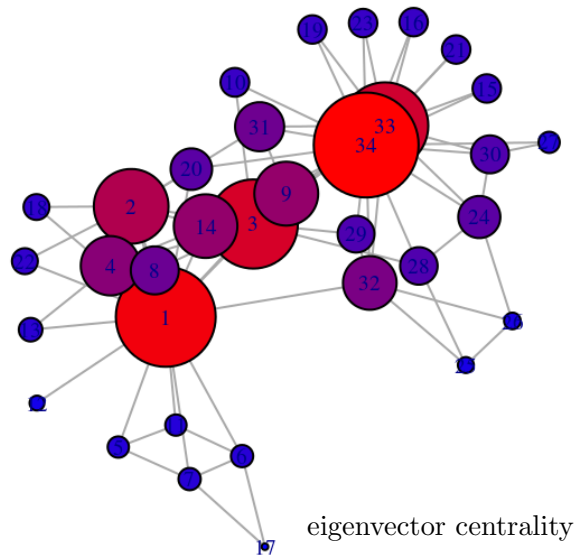
degree centrality



closeness centrality



betweenness centrality



eigenvector centrality

source: D. Petrov, Y. Dodonova, A. Shestakov (2015)

Trustworthy Webpages

We study two scoring methods (both based on voting from neighbors) that are more suitable to webgraphs

- Pagerank
- HITS

Node Centrality in Graphs

$N^+(p)$ ($N^-(p)$): out(in)-neighbors

$d^+(p)$ and $d^-(p)$: out(in)-degree

- Rating based on out-degree
  - Very easy to inflate
- Rating based on in-degree
  - Can be manipulated by spam farm many interlinked *'fake'* pages
- Rating based on weighted in-degree
  - Weights would be ratings of in-linking pages
  - Compare rating of p_1 and p_2 each with 2 in-links from p_x and p_y
  - Suppose p_x and p_y are equally rated
  - What if p_x links to 1000 other pages and p_y only links to p_2

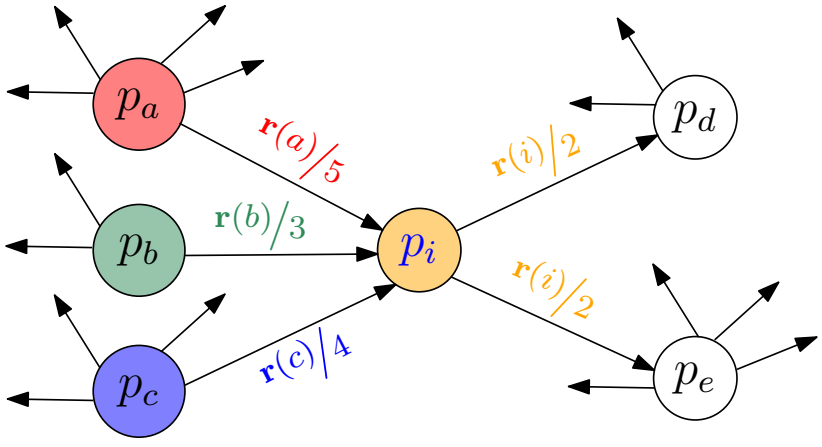
Pagerank

- Score page p_i based on *'weighted voting'* of in-neighbors

For weights consider

- rank of in-neighbors
- importance of incoming links

$$\mathbf{r}(p_i) = \sum_{p_j \in N^-(p_i)} \frac{\mathbf{r}(p_j)}{d^+(p_j)}$$



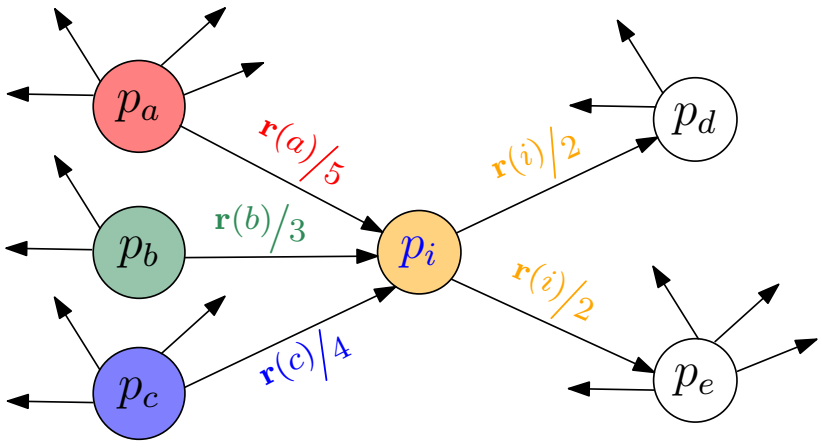
- Each page divides its score equally among its out-neighbors
- Rating of a page is directly proportional to rating of pages linking to it

Pagerank

For weights consider

- rank of in-neighbors
- importance of incoming links

$$\mathbf{r}(p_i) = \sum_{p_j \in N^-(p_i)} \frac{\mathbf{r}(p_j)}{d^+(p_j)}$$



- Each page divides its score equally among its out-neighbors
- Rating of a page is directly proportional to rating of pages linking to it
- Recursive Formulation!

$\mathbf{r}^{(0)}(p_i) \leftarrow 1/n$ ▷ At time $t = 0$ give each page equal rating

$$\mathbf{r}^{(t+1)}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \frac{\mathbf{r}^{(t)}(p_j)}{d^+(p_j)}$$ ▷ repeated improvement at time t

Pagerank Algorithm

Algorithm Pagerank

$\mathbf{r}^{(0)}(p_i) \leftarrow 1/n$ $\quad \triangleright\ n = |V(\text{WEB})|$

for $t = 0$ to k **do**

$$\mathbf{r}^{(t+1)}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \frac{\mathbf{r}^{(t)}(p_j)}{d^+(p_j)}$$

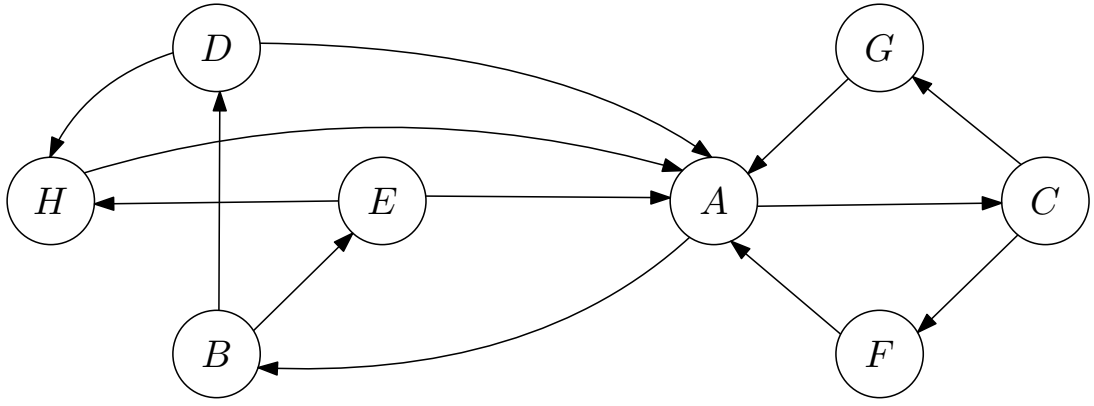
- Uses principle of repeated improvement
- Total pagerank (sum over all pages) remains constant $= 1$

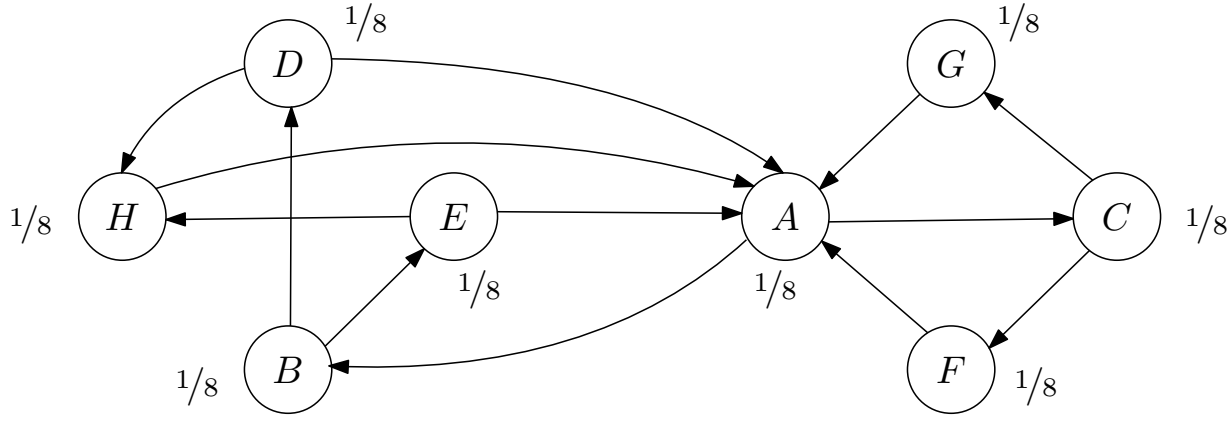
Pagerank Algorithm

$r^{(0)}(p_i) \leftarrow 1/n$ ▷ $n = |V(\text{WEB})|$

for $t = 0$ to k **do**

$r^{(t+1)}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \frac{r^{(t)}(p_j)}{d^+(p_j)}$





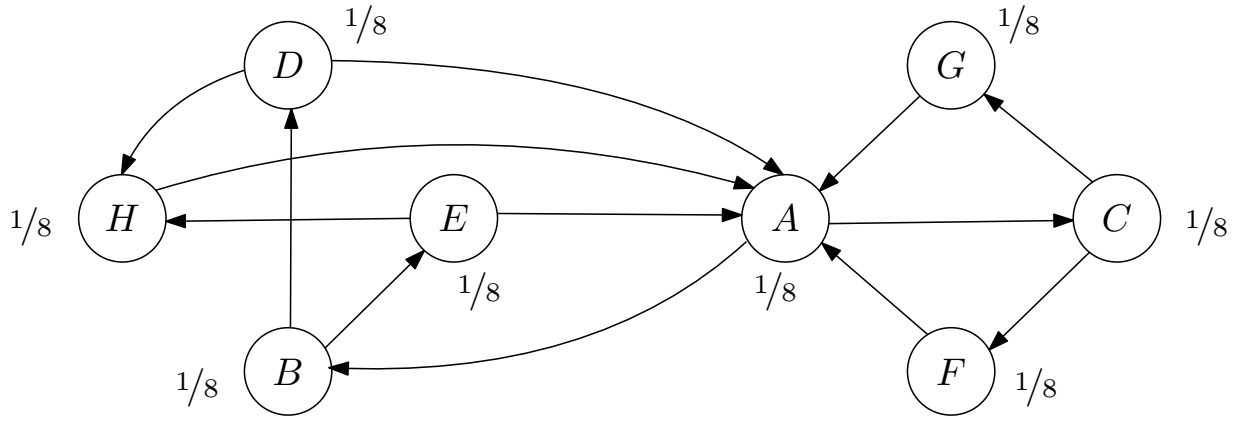
node	$\mathbf{r}(\cdot)$
A	1/8
B	1/8
C	1/8
D	1/8
E	1/8
F	1/8
G	1/8
H	1/8

Pagerank Algorithm

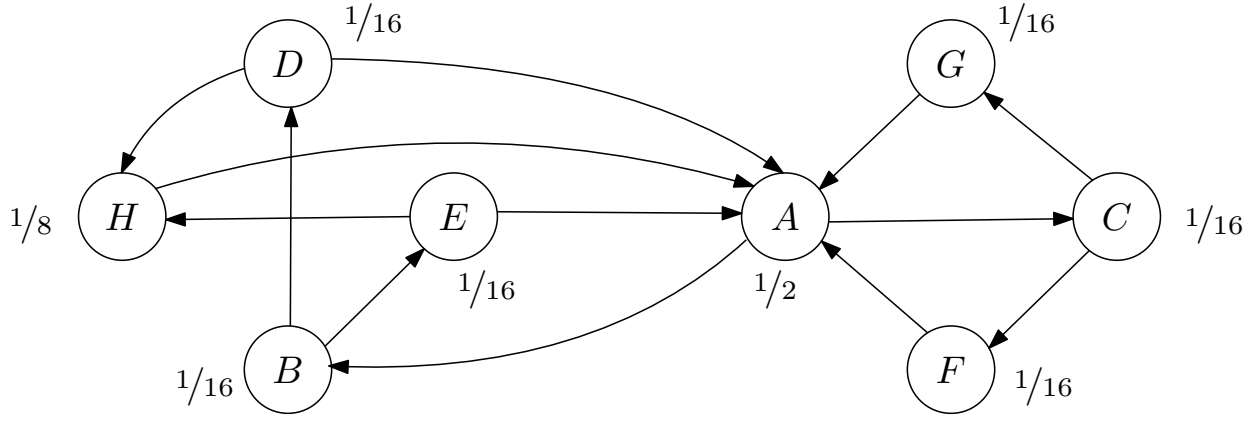
$r^{(0)}(p_i) \leftarrow 1/n$ ▷ $n = |V(\text{WEB})|$

for $t = 0$ to k **do**

$r^{(t+1)}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \frac{r^{(t)}(p_j)}{d^+(p_j)}$



node	$\mathbf{r}(\cdot)$
A	1/8
B	1/8
C	1/8
D	1/8
E	1/8
F	1/8
G	1/8
H	1/8



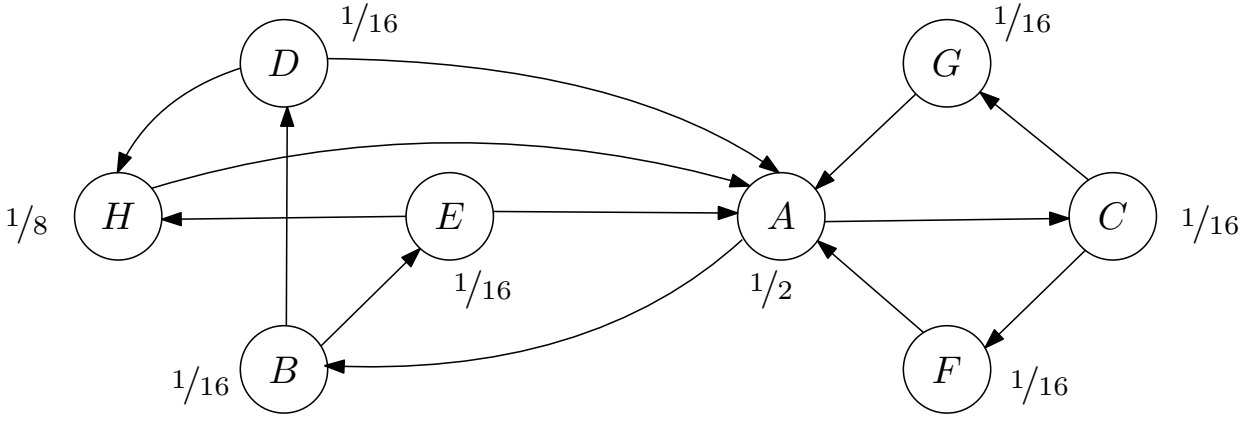
node	$\mathbf{r}(\cdot)$
A	1/2
B	1/16
C	1/16
D	1/16
E	1/16
F	1/16
G	1/16
H	1/8

Pagerank Algorithm

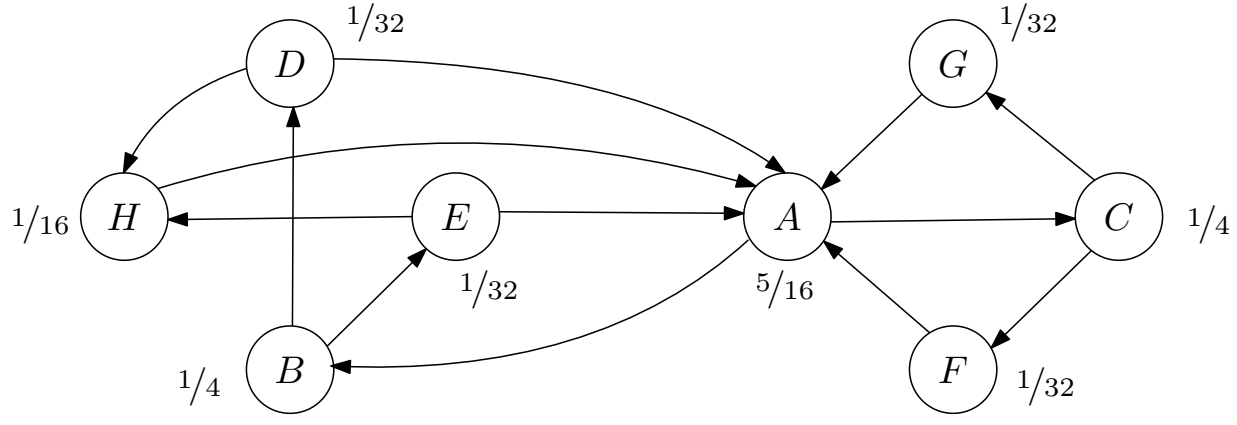
$r^{(0)}(p_i) \leftarrow 1/n$ ▷ $n = |V(\text{WEB})|$

for $t = 0$ to k **do**

$r^{(t+1)}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \frac{r^{(t)}(p_j)}{d^+(p_j)}$



node	$\mathbf{r}(\cdot)$
A	1/2
B	1/16
C	1/16
D	1/16
E	1/16
F	1/16
G	1/16
H	1/8



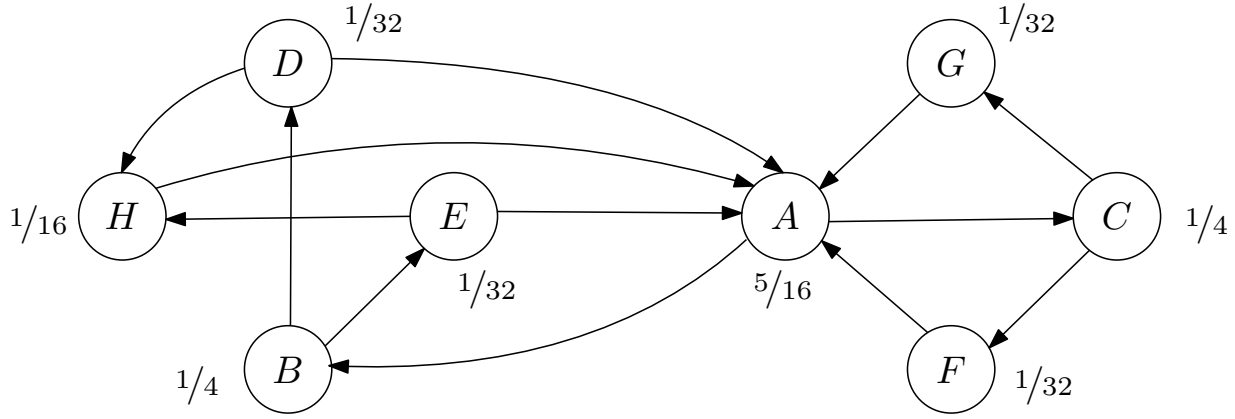
node	$\mathbf{r}(\cdot)$
A	5/16
B	1/4
C	1/4
D	1/32
E	1/32
F	1/32
G	1/32
H	1/16

Pagerank Algorithm

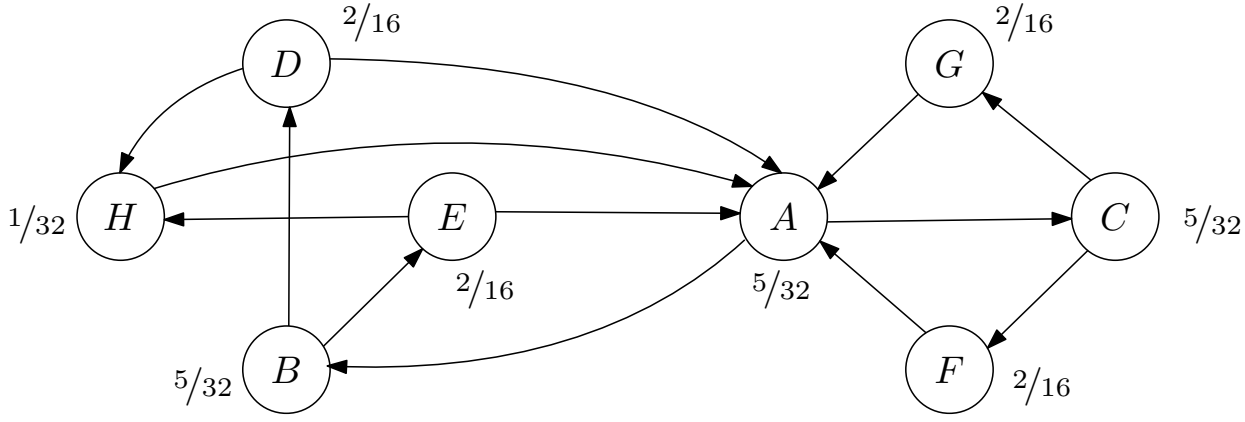
$r^{(0)}(p_i) \leftarrow 1/n$ ▷ $n = |V(\text{WEB})|$

for $t = 0$ to k **do**

$r^{(t+1)}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \frac{r^{(t)}(p_j)}{d^+(p_j)}$



node	$\mathbf{r}(\cdot)$
A	5/16
B	1/4
C	1/4
D	1/32
E	1/32
F	1/32
G	1/32
H	1/16



node	$\mathbf{r}(\cdot)$
A	5/32
B	5/32
C	5/32
D	2/16
E	2/16
F	2/16
G	2/16
H	1/32

Pagerank Algorithm

Algorithm Pagerank

$\mathbf{r}^{(0)}(p_i) \leftarrow 1/n$ ▷ $n = |V(\text{WEB})|$

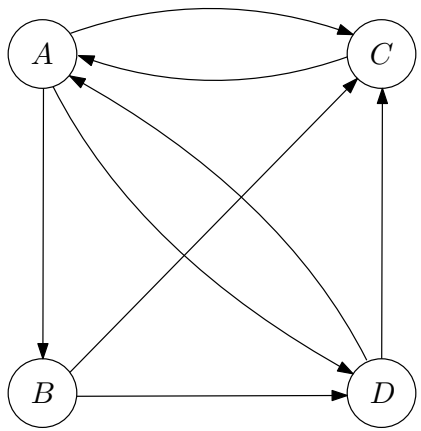
for $t = 0$ to **k** **do** ▷ What is k?

$$\mathbf{r}^{(t+1)}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \frac{\mathbf{r}^{(t)}(p_j)}{d^+(p_j)}$$

- Uses principle of repeated improvement
- No such k, stopping condition is $\|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}\| < \epsilon$ for $0 < \epsilon < 1$
- This approach is the power iteration method to compute eigenvector
- If the graph is not a degenerate case, the pagerank values converge to a limiting vector (equilibrium, stationary distribution)
- Total pagerank (sum over all pages) remains constant $= 1$

Random Walk Formulation of Pagerank

- Simulate a random walk (random surfer) across the web digraph
- The surfer chooses an outgoing link at random
- Score of a page is the (long-term) probability of visiting it



Link Matrix, L

Transpose of out-degree normalized adjacency matrix

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

- L encodes probabilities of vising a page from another (transition)

$$\mathbf{r}(p_i) = \sum_{p_j \in N^-(p_i)} \frac{\mathbf{r}(p_j)}{d^+(p_j)} \qquad \mathbf{r}(p_i) = \sum_{p_j \in V} L(i,j)\, \mathbf{r}(p_j)$$

- $\mathbf{r} = L\mathbf{r}$ (the eigenvector of L with eigenvalue 1)
- $\mathbf{r} = L\mathbf{r}$ (also the stationary distribution of the Markov chain L)

Pagerank Algorithm

Algorithm Pagerank

$\mathbf{r}^{(0)}(p_i) \leftarrow 1/n$ $\quad \triangleright\ n = |V(\text{WEB})|$

while $\|\mathbf{r}^{(t)} - \mathbf{r}^{(t-1)}\| > \epsilon$ **do**

$$\mathbf{r}^{(t+1)}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \frac{\mathbf{r}^{(t)}(p_j)}{d^+(p_j)}$$

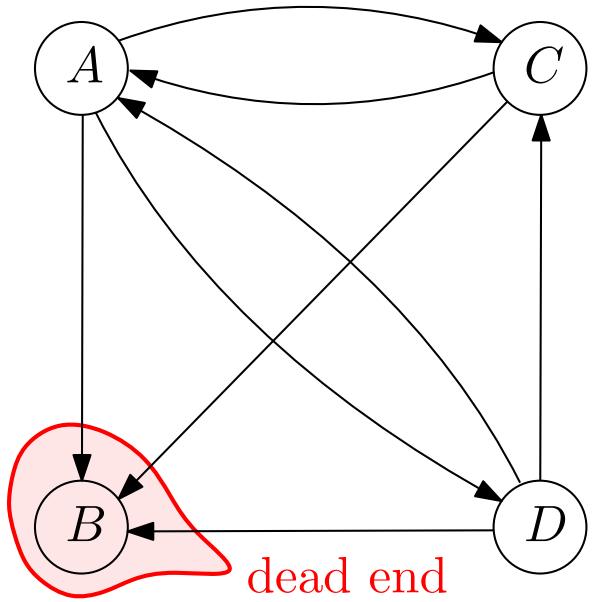
- Does it always converge to a unique and meaningful solution?

Fundamental problems

- Dangling node or dead ends
  - Node(s) with out-degree 0 (sink nodes)
  - Since rating is not distributed, total rank leaks out
- Spider Traps
  - All out-links within a component (sink components)
  - They eventually absorb all the rating
- Both above make the graph not strongly connected

Pagerank Algorithm: Dead Ends

- Dead ends: Sink nodes
- Total rank leaks out



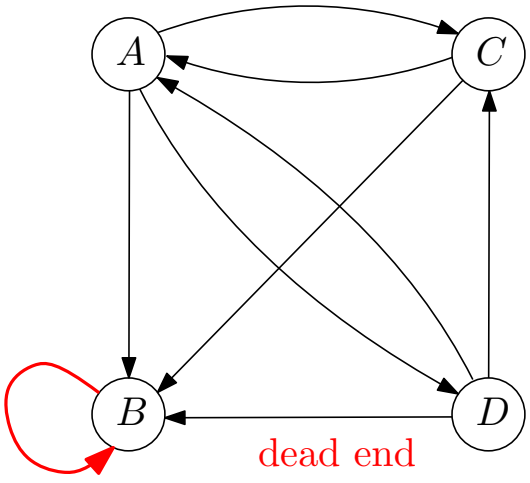
$$\begin{bmatrix} 0 & 0 & 1/2 & 1/3 \\ 1/3 & 0 & 1/2 & 1/3 \\ 1/3 & 0 & 0 & 1/3 \\ 1/3 & 0 & 0 & 0 \end{bmatrix}$$

Link Matrix, L

node	$\mathbf{r}^{(0)}(\cdot)$	$\mathbf{r}^{(1)}(\cdot)$	$\mathbf{r}^{(2)}(\cdot)$	$\mathbf{r}^{(3)}(\cdot)$
A	0.25	0.2083	0.1111	0.0718
B	0.25	0.2917	0.1806	0.1088
C	0.25	0.1667	0.0972	0.0602
D	0.25	0.0833	0.0694	0.0370
Total pagerank = 1.0		0.75	0.4583	0.2778

Pagerank Algorithm: Dead Ends

- Make a self-loop from a dangling node to itself
- This benefits such pages, become spider traps



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

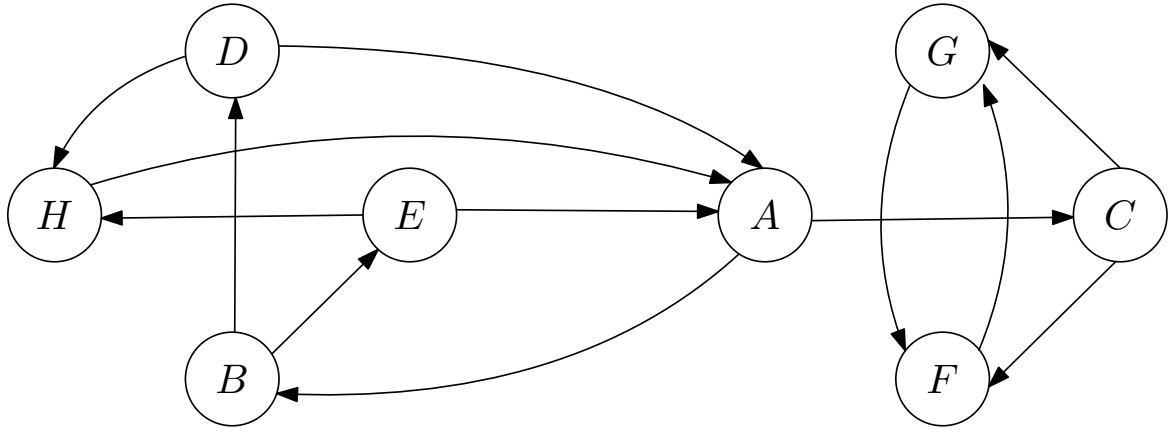
Link Matrix, L

node	$\mathbf{r}^{(0)}(\cdot)$	$\mathbf{r}^{(1)}(\cdot)$	$\mathbf{r}^{(2)}(\cdot)$	$\cdots$	$\mathbf{r}^{(*)}(\cdot)$
A	0.25	0.2083	0.1111		0.0
B	0.25	0.5417	0.7222		1.0
C	0.25	0.1667	0.0972		0.0
D	0.25	0.0833	0.0694		0.0

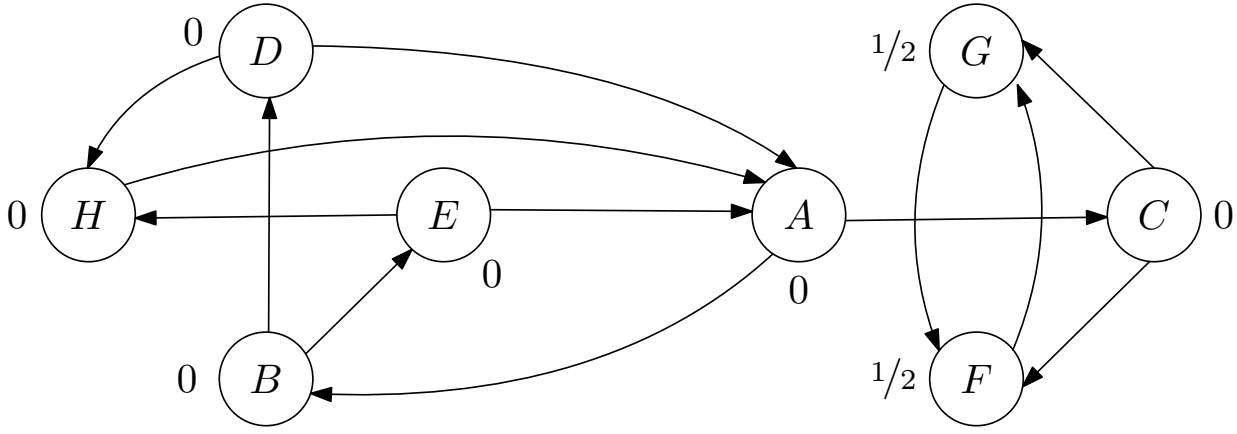
- A better solution is to recursively prune them out
- Compute pagerank for the remaining nodes
- 'pass them on to the pruned-out pages'
- Other methods to deal with them are also used (see below)

Pagerank Algorithm: Spider Traps

- Spider Traps: sink component(s)
- Eventually absorb all the rating



node	$\mathbf{r}(\cdot)$
A	1/8
B	1/8
C	1/8
D	1/8
E	1/8
F	1/8
G	1/8
H	1/8

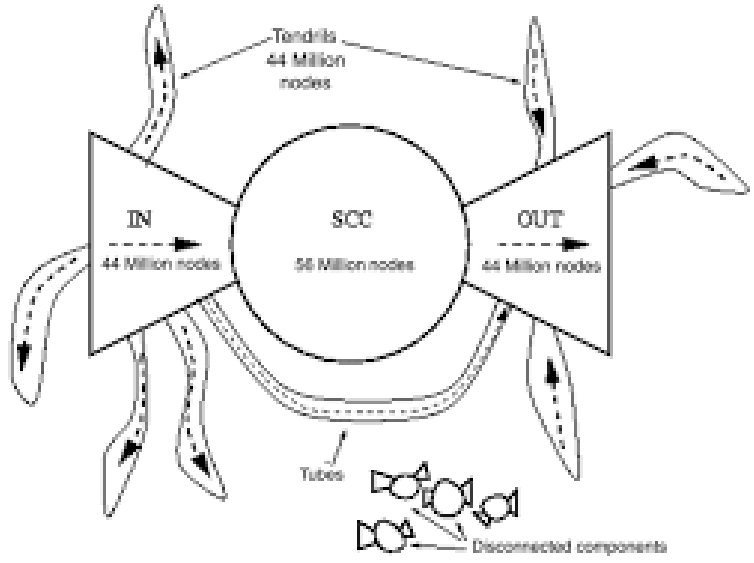


node	$\mathbf{r}(\cdot)$
A	0
B	0
C	0
D	0
E	0
F	1/2
G	1/2
H	0

Pagerank Algorithm: Web Structure

- Structural challenges to Web IR algorithms are not only hypothetical
- Broder et.al. (2000) SCC analysis of webgraph (AltaVista index)
- Study replicated for larger recent webgraphs reveal similar structure

- $\sim 200m$ pages, $\sim 1.5b$ links
- bow-tie structure (macroscopic)
- grouping of SCC's
- CORE: a giant SCC ($\sim 56m$) nodes
- IN: can reach CORE (unidirectional)
- OUT: can be reached from CORE
- TENDRILS:
  - reachable from IN cannot reach CORE
  - can reach OUT not reachable from CORE
- TUBES: both types of tendrils
- DISCONNECTED COMPONENTS



The bow-tie structure of the web (A. Broder et.al (2000))

Pagerank Algorithm: Random Teleports

- Spider Traps: sink component(s)
- Eventually absorb all the rating
- Google fix is "random restarts" (random teleport)
  - With probability $1 - \beta$ follow an out-link
  - With probability β jump to a random page
  - Generally, $\beta \in [0.1 - 0.15]$
- Random walker (surfer) will teleport out of a spider trap
- From dead-ends teleport with probability 1
  - Matrix becomes column-stochastic

$$\mathbf{r}(p_i) = \sum_{p_j \in N^-(p_i)} (1-\beta)\frac{\mathbf{r}(p_j)}{d^+(p_j)} + \beta\frac{1}{n}$$

Adjusted Formulation

$$L' = (1-\beta)L + \frac{\beta}{n}\mathbb{I}_n$$

Web Information Retrieval: Link Spamming

- Link Farms: Densely connected subgraphs to increase ranks of pages
- Private Blog Networks: authoritative expired websites with influential in-links used to have out-links to targeted web pages
- Sybil Attack: spammer create multiple inter-linked websites at different domain names
- Spam Blogs: Blogs created for promoting a webpage
- Guest blog Spam:
- Referrer Log Spamming:
- Forum Spam, Comment Spam, Wiki Spam

Anchor Text

Google penalizes various content and link spamming tricks (Google Panda)

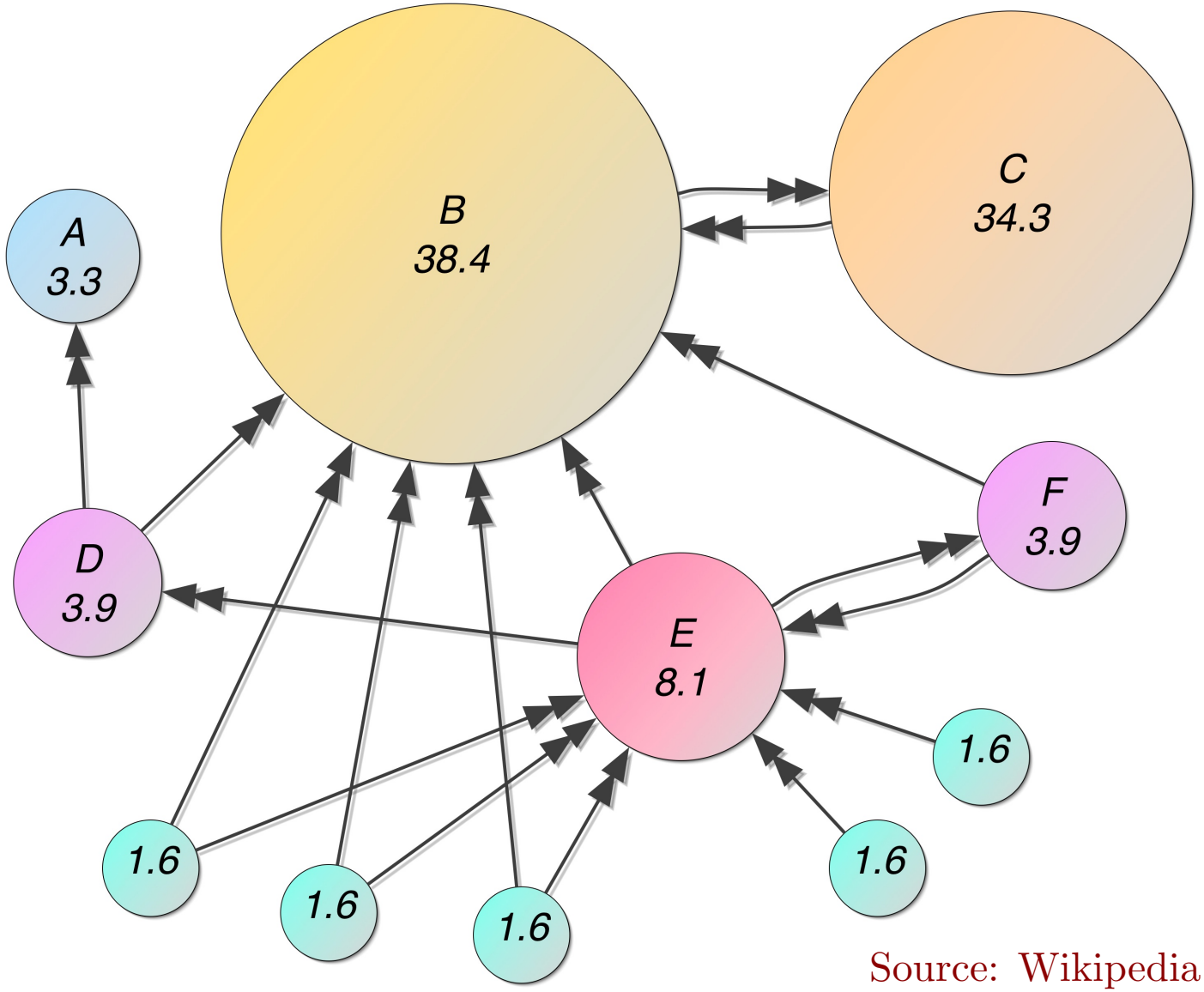
Anchor text, link label or link text is the visible, clickable text in hyperlinks

Links types: No anchor text (click here), Naked URL (www.abc.com) or (LUMS)

- (Legitimate) Anchor text describes the landing page better than the content of the page itself
- Anchor texts from links pointing to a page is included when the page is indexed
- Terms from anchor text are weighted highly in the vector representation of the page
- Help index non-html or non-text pages too (images/videos)

A related concept is that of Google Bomb

Pagerank Visualization



Source: Wikipedia

Visualization of pageranks (percentages) for a small graph
damping factor, $\beta = .85$

Personalized and Topic Sensitive Pagerank

- So far we discussed query independent ranking
  - Compute an apriori rating $\mathbf{r}$ of all web pages in the index
  - On query q, find the subset C of pages relevant to q
  - Present pages in C in decreasing order of $\mathbf{r}$
- Specialized Ranking w.r.t query
  - On query q by user u
  - Personalized Pagerank [Brin & Page 1998] adjust ordering according to the user u
  - Topic Sensitive PageRank [Taher Haveliwala 2002] adjust rating according to topic of the query q

Topic Sensitive Pagerank

Goal: Not just PageRank - rate pages also by relevance to topic of query q

In conventional PageRank we teleported to any of the pages equally likely

$$\mathbf{r}(p_i) = \sum_{p_j \in N^-(p_i)} (1-\beta)\frac{\mathbf{r}(p_j)}{d^+(p_j)} + \beta \underbrace{\frac{1}{n}}_{T}$$

T is a uniform distribution over all pages

Topic sensitive PageRank

- identify a subset of pages S (how?) ▷ teleport set
- related to the topic of q (what is topic of q?)

T_S: non-uniform probability distribution (high values at coordinates in S)

Random walker is biased towards S more likely to jump onto pages in S (hence spend more times), their ratings will be higher

Can we compute PageRank at query time?

Topic Sensitive Pagerank

Preprocessing:

- Fix a set of k topics
- Find pages about each topics (k teleport sets $S_1, S_2, \ldots, S_k$)
- For each topic find PageRank of all pages using T_{S_i} for teleporting
  - Each page has k PageRank scores, $\mathbf{r}_1(\cdot), \ldots, \mathbf{r}_k(\cdot)$ - one for each topic

Query-time processing:

- Compute distribution of likelihoods of q belonging to each of k topics
  - i.e. for $1 \leq i \leq k$ find $Pr[C_i|q]$ (probability that q's topic is C_i)
- TSPR of page u is weighted (by $Pr[C_i|q]$) sum of $\mathbf{r}_i(u)$

$$TSPR(u|q) = \sum_{i=1}^{k} Pr[C_i|q]\, \mathbf{r}_i(u)$$

Topic Sensitive Pagerank

Preprocessing:

- Fix a set of k topics
- Find pages about each topics (k teleport sets $S_1, S_2, \ldots, S_k$)
- DMOZ top level, (sports, business, health, education)
- Use topic modeling, cluster TF-IDF vectors of pages, use document embedding and deep learning

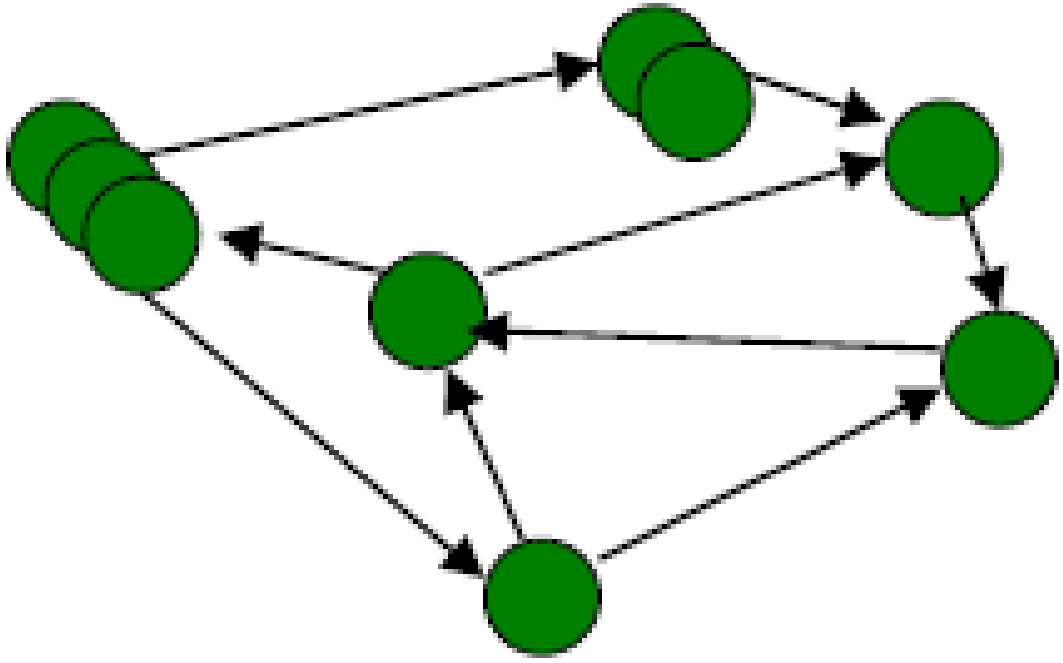
Topic Sensitive Pagerank

Query-time processing:

- Compute distribution of likelihoods of q belonging to each of k topics
  - User can pick topic from a combo-box
  - Use classification to classify query into a topic
  - Use query launching context
    - e.g. query launched from a webpage (local search bar) about topic i
    - History of queries e.g. "basketball" followed by "Jordan"
  - Use user context e.g. Users social media profile, attributes etc.

Topic Sensitive Pagerank

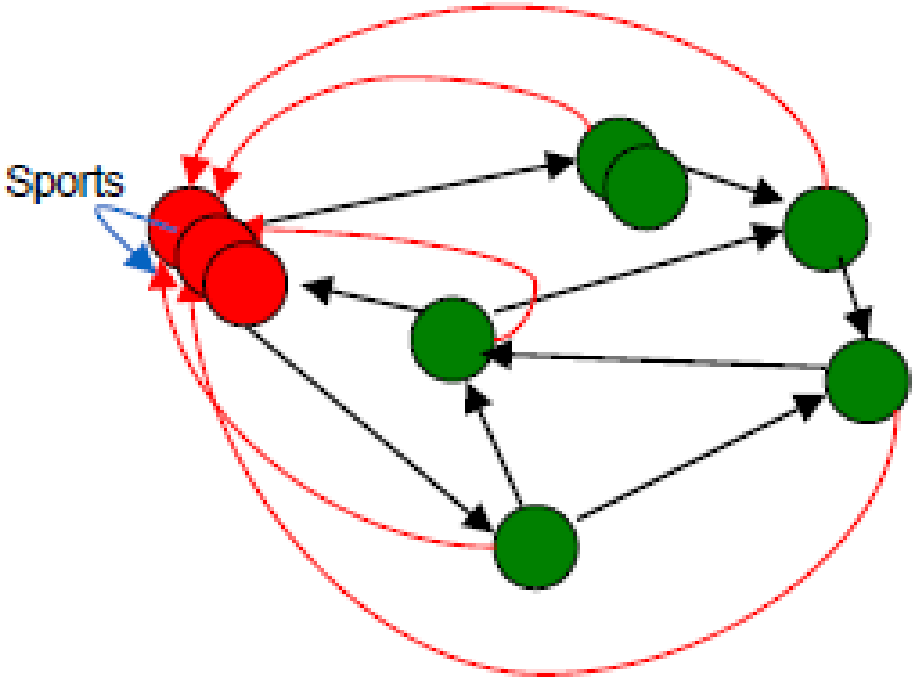
J. Magalhaes @Universidade NOVA de Lisboa



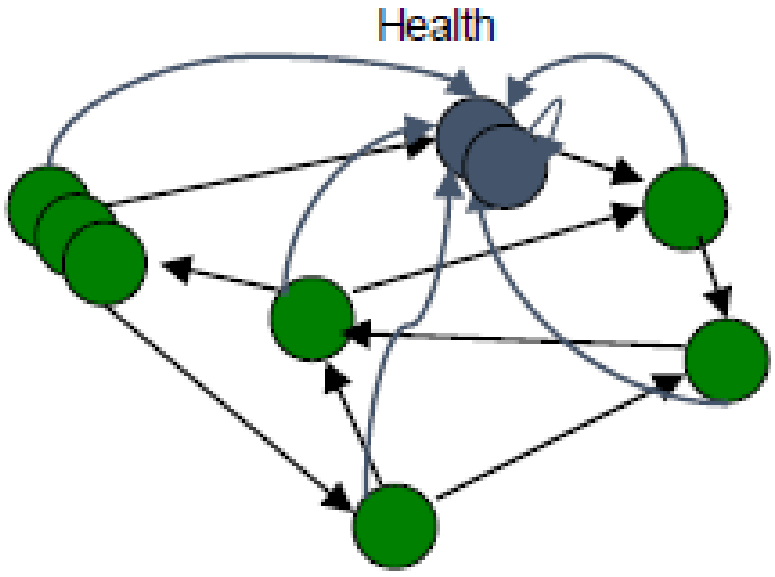
The query has 90% chance of being about **Sports**.

The query has 10% chance of being about **Health**.

Topic Sensitive Pagerank

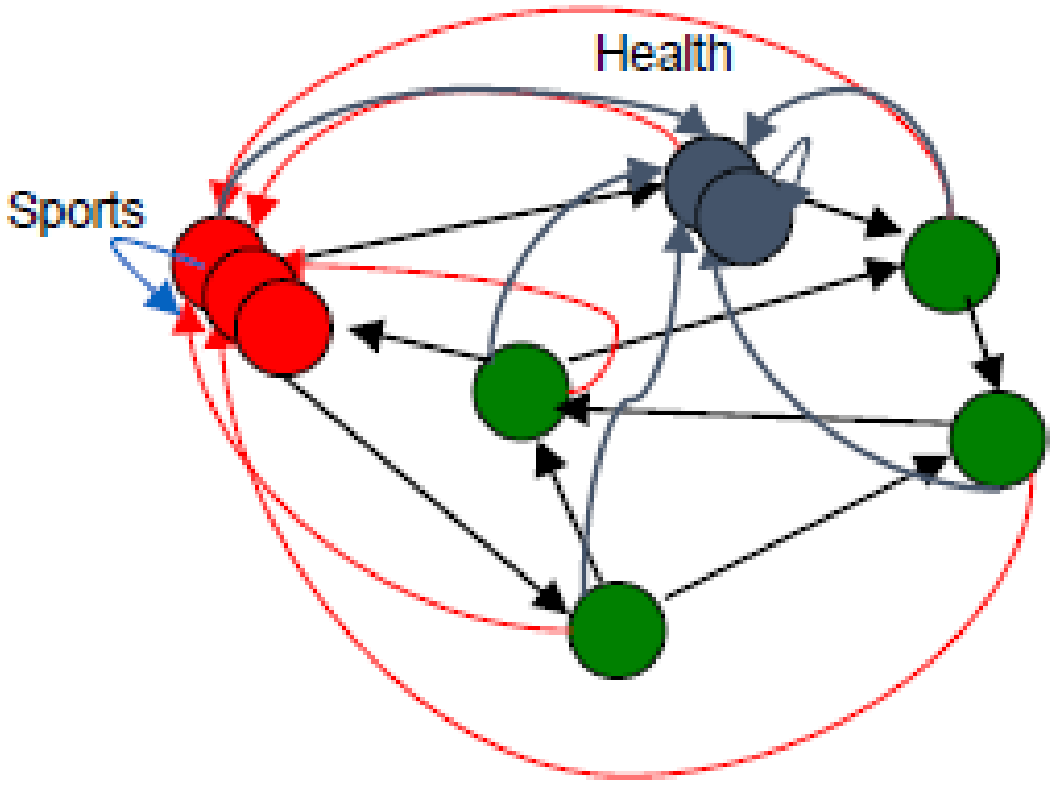


Sports teleportation



Health teleportation

Topic Sensitive Pagerank



pr = (0.9 PR_{sports} + 0.1 PR_{health}) gives you:
9% sports teleportation, 1% health teleportation

Hyperlink-Induced Topic Search (HITS)

Kleinberg (1998)

- PageRank: best suited for most reliable pages to specific queries
- HITS: best suited for “broad topic“ queries
- It returns a broader common opinion
- Not only find pages that reliably has relevant content but also finds “experts” on the topic, pages linking to many relevant pages
- In response to a query HITS finds two sets of inter-related pages

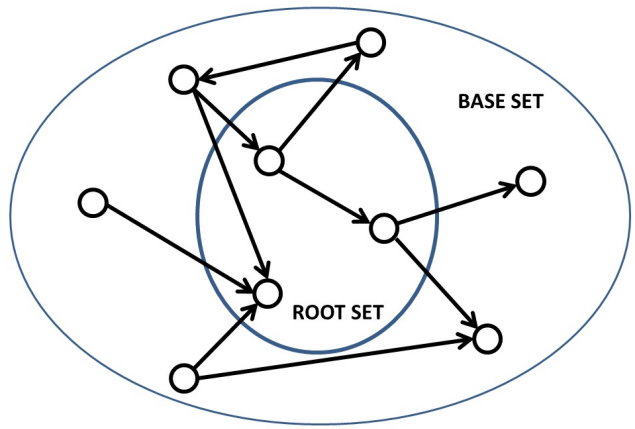
Hyperlink-Induced Topic Search (HITS)

Underlying Assumption: Page A links to Page $B \implies A$ recommends B

- Hubs (high quality experts): pages with list of "good pages"
  - List of top data science conferences
  - Course bulletin
- Authorities (high quality content): pages listed on many "good hubs"
  - Conference webpages
  - Course home pages

Hyperlink-Induced Topic Search (HITS)

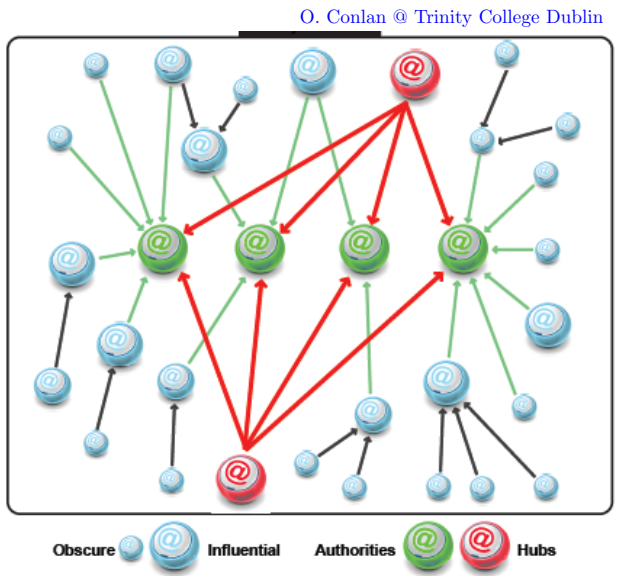
- Extract from the index a subset of pages that could be potentially good hubs or authorities (base set, B) as follows
  - On query q, get root set: of pages most "relevant" to q
  - Add pages that are either
    - linking to a page in the root set, or
    - linked to by a page in the root set



- For each page in B, compute its hub rating and authority rating
- Return the top k hubs and authorities from the base set

Hyperlink-Induced Topic Search (HITS)

- Each page x (in the base set) has two scores
  - $\mathbf{h}(x)$: hub score of x : measure quality of x as an "expert"
  - $\mathbf{a}(x)$: authority score of x : measure quality of x as "content"
- Initialize $\mathbf{h}(x)$ and $\mathbf{a}(x)$ to 1 for all x
- Principle of repeated improvement
- A page is a good authority if it is linked to by good hubs
- A page is a good hub if it links to good authorities



Hyperlink-Induced Topic Search (HITS)

- Each page x (in the base set) has two scores
  - $\mathbf{h}(x)$: hub score of x : measure quality of x as an "expert"
  - $\mathbf{a}(x)$: authority score of x : measure quality of x as "content"
- Initialize $\mathbf{h}(x)$ and $\mathbf{a}(x)$ to 1 for all x
- Principle of repeated improvement
- A page is a good authority if it is linked to by good hubs
- A page is a good hub if it links to good authorities
- $\mathbf{a}(x)$ is sum of hub scores of pages pointing to x

$$\mathbf{a}(x) \leftarrow \sum_{y \in N^-(x)} \mathbf{h}(y)$$

- $\mathbf{h}(x)$ is sum of authority scores of pages x is pointing to

$$\mathbf{h}(x) \leftarrow \sum_{y \in N^+(x)} \mathbf{a}(y)$$

HITS Algorithm

Hyperlink-Induced Topic Search (HITS)

Algorithm HITS

$\mathbf{h} \leftarrow \textsc{ones}(|B|)$ ▷ B : base set

$\mathbf{a} \leftarrow \textsc{ones}(|B|)$ ▷ Initialize $\mathbf{h}(\cdot)$ and $\mathbf{a}(\cdot)$ to 1

while stopping condition is not met **do**

$\textsc{normalize}(\mathbf{a}, \mathbf{h})$ ▷ $\mathbf{h} \leftarrow \mathbf{h}/\|\mathbf{h}\|$

For each $p_i \in B$

$$\mathbf{h}(p_i) \leftarrow \sum_{p_j \in N^+(p_i)} \mathbf{a}(p_j)$$

$$\mathbf{a}(p_i) \leftarrow \sum_{p_j \in N^-(p_i)} \mathbf{h}(p_j)$$

HITS Algorithm

- Each page i (in the base set) has two scores
  - $\mathbf{h}(i)$: hub score of i: measure quality of i as an "expert"
  - $\mathbf{a}(i)$: authority score of i: measure quality of i as "content"
- Initialize $\mathbf{h}(i)$ and $\mathbf{a}(i)$ to 1 for all i

$$\mathbf{a}(i) = \sum_{j \in N^-(i)} \mathbf{h}(j) \qquad \mathbf{h}(i) = \sum_{j \in N^+(i)} \mathbf{a}(j)$$

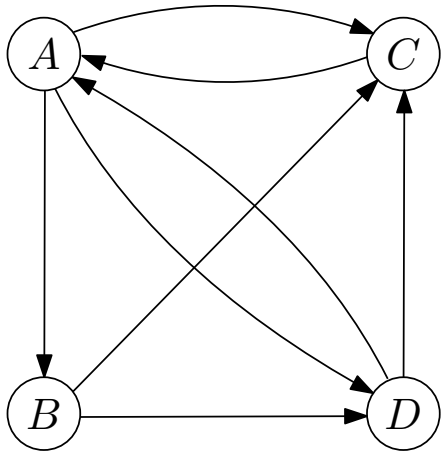
Let A be the adjacency matrix of the subgraph induced by B, $|B| = n$

A is $n \times n$ matrix with $A(i,j) = 1$ if $p_i \rightarrow p_j$

$$\mathbf{h}(i) = \sum_{j \in N^+(i)} \mathbf{a}(j) \quad \Leftrightarrow \quad \mathbf{h}(i) = \sum_j A(i,j)\, \mathbf{a}(j) \quad \Leftrightarrow \quad \mathbf{h} = A\mathbf{a}$$

$$\mathbf{a}(i) = \sum_{j \in N^-(i)} \mathbf{h}(j) \quad \Leftrightarrow \quad \mathbf{a}(i) = \sum_j A(j,i)\, \mathbf{h}(j) \quad \Leftrightarrow \quad \mathbf{a} = A^T\mathbf{h}$$

HITS Algorithm



Adjacency Matrix

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{h} = A\,\mathbf{a}$$

$$\mathbf{a} = A^T\,\mathbf{h}$$

node	$\mathbf{h}^{(0)}(\cdot)$	$\mathbf{h}^{(1)}(\cdot)$	$\mathbf{h}^{(2)}(\cdot)$	$\mathbf{h}^{(3)}(\cdot)$	$\mathbf{h}^{(4)}(\cdot)$	$\mathbf{h}^{(5)}(\cdot)$
A	1	3	6	15	33	79
B	1	2	5	12	27	64
C	1	1	2	3	7	13
D	1	2	5	10	23	50

node	$\mathbf{a}^{(0)}(\cdot)$	$\mathbf{a}^{(1)}(\cdot)$	$\mathbf{a}^{(2)}(\cdot)$	$\mathbf{a}^{(3)}(\cdot)$	$\mathbf{a}^{(4)}(\cdot)$	$\mathbf{a}^{(5)}(\cdot)$
A	1	2	3	7	13	30
B	1	1	3	6	15	33
C	1	3	7	16	37	83
D	1	2	5	11	27	60

HITS Algorithm

Algorithm HITS

$\mathbf{h} \leftarrow \text{ONES}(|B|)$

$\mathbf{a} \leftarrow \text{ONES}(|B|)$

while stopping condition is not met **do**

$\text{NORMALIZE}(\mathbf{a}, \mathbf{h})$

$\mathbf{h} \leftarrow A\,\mathbf{a}$

$\mathbf{a} \leftarrow A^T\mathbf{h}$

$$\mathbf{h} = A\,\mathbf{a} = A\,\underbrace{A^T\mathbf{h}} = \underbrace{A\,A^T}\mathbf{h} \qquad \mathbf{a} = A^T\mathbf{h} = A^T\,\underbrace{A\mathbf{a}} = \underbrace{A^T A}\mathbf{a}$$

In $2k$ steps $\quad \mathbf{h} = (A\,A^T)^k\mathbf{h} \quad \mathbf{a} = (A^T A)^k\mathbf{a}$

- Under some reasonable conditions on A, HITS converges to $\mathbf{h}^*$ and $\mathbf{a}^*$
- $\mathbf{h}^*$ is the principal eigenvector of AA^T
- $\mathbf{a}^*$ is the principal eigenvector of $A^T A$

HITS Algorithm

- Google uses a very complicated algorithm, incorporating more than a 100 factors
- The current Google algorithm is very close to HITS
- HITS is implemented in Ask.com and teoma