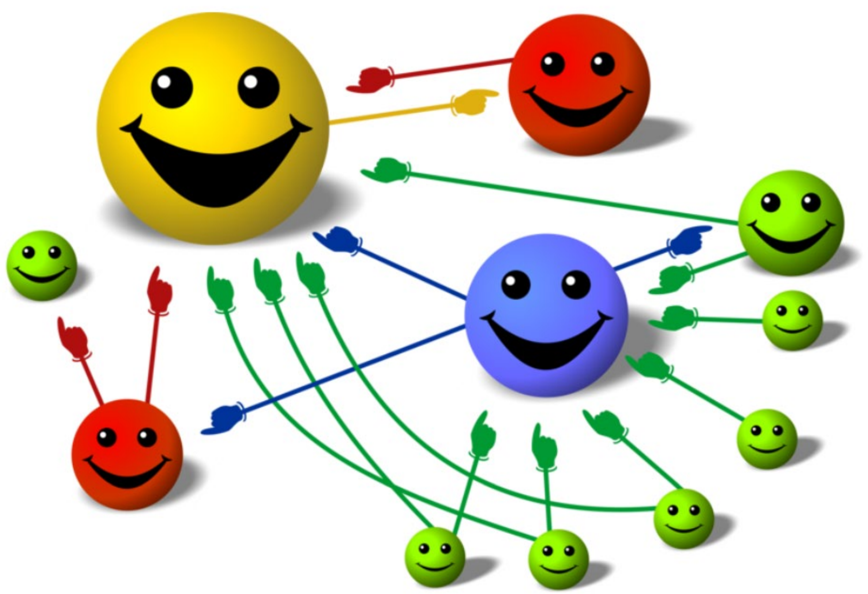
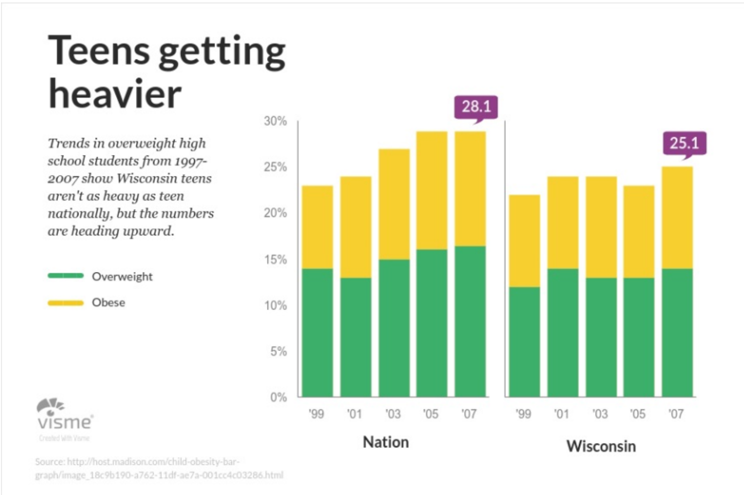
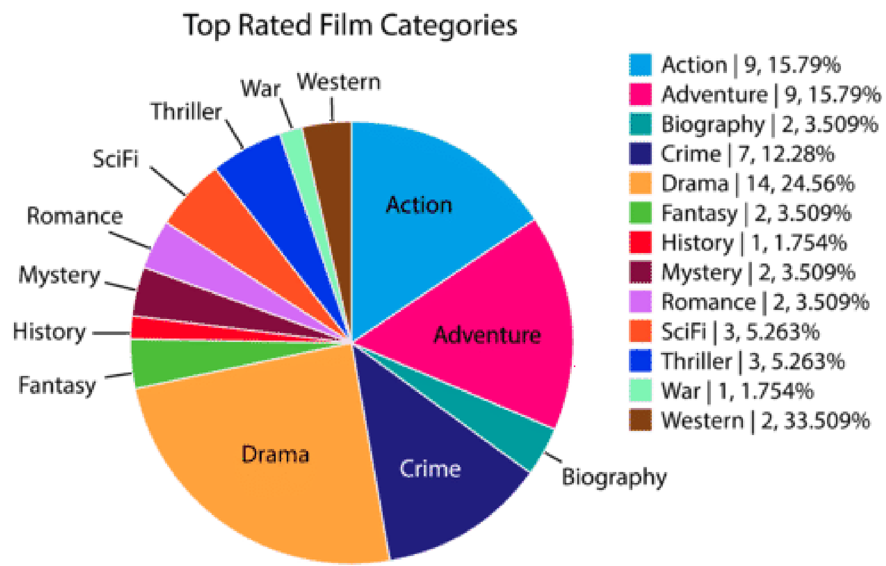
CS5344
Analyzing Large Graphs

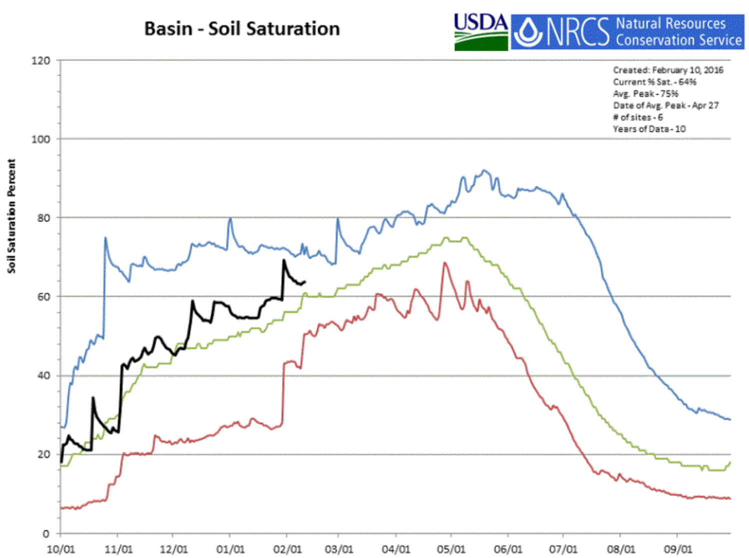


Acknowledgement: Slides adapted from http://www.mmds.org

What is a Graph?





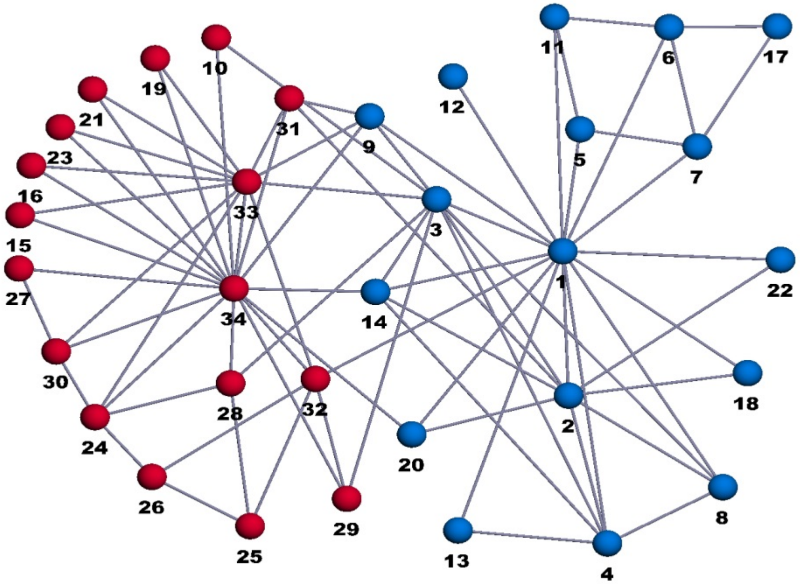


These are charts, NOT graphs!

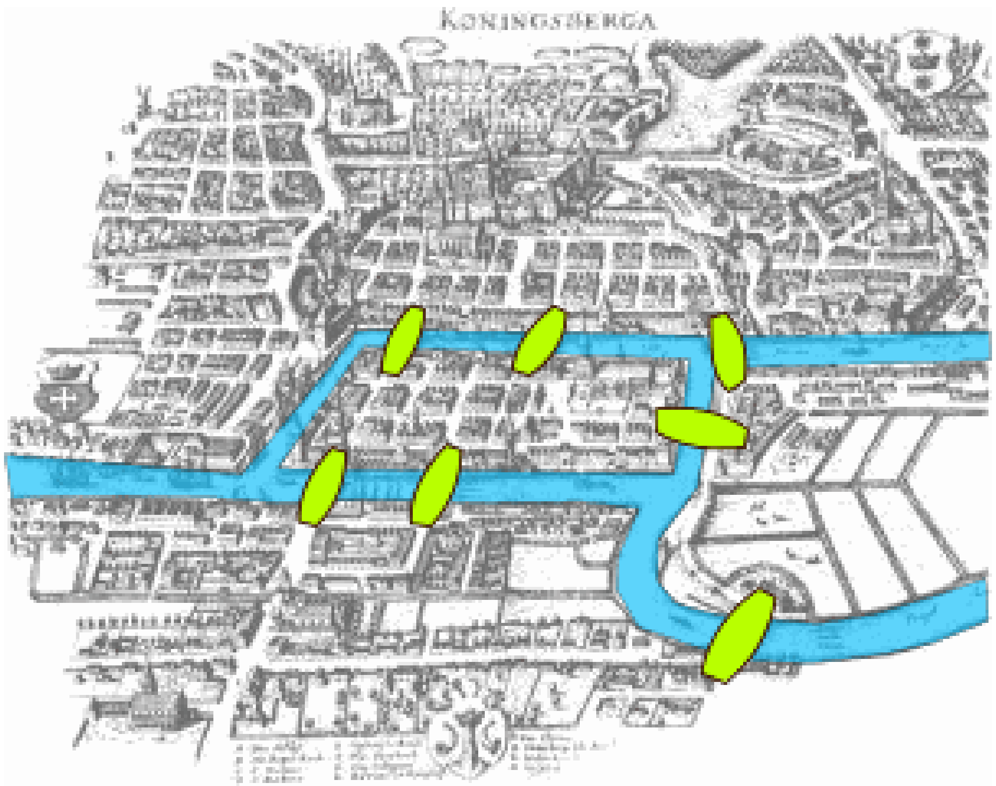
What is a Graph?

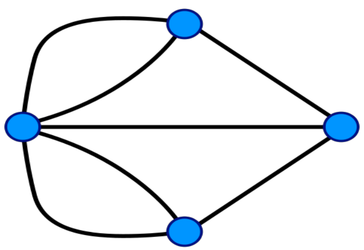
- Objects (nodes): {1, 2, 3, ...}
- Relationships (edges): { {1,2}, {2,3}… }
- Graph analytics aim to understand and visualize **relationships that exist between objects**, to uncover **insights about the structures and patterns** of objects relationships

Community structure in the social network of Zachery Karate Club



Example: Technological Networks



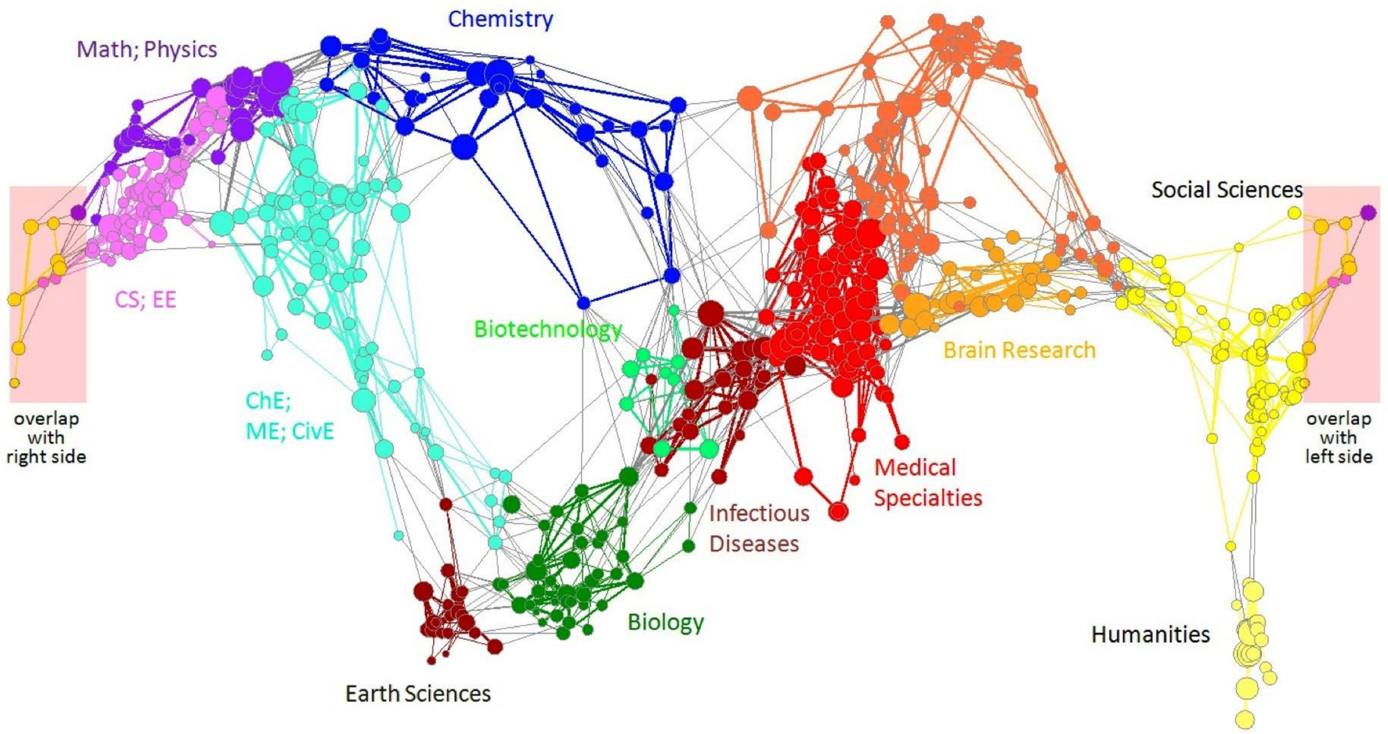


Seven Bridges of Königsberg

[Euler, 1735]

- **Return to the starting point by traveling each link of the graph once and only once.**

Example: Information Nets



Citation Networks and Maps of Science

[Borner et. al, 2012]

- **Links denote citations between papers in journal**
- **Visualize how different disciplines relate to each other**

Example: Social Networks

Facebook Social Graph

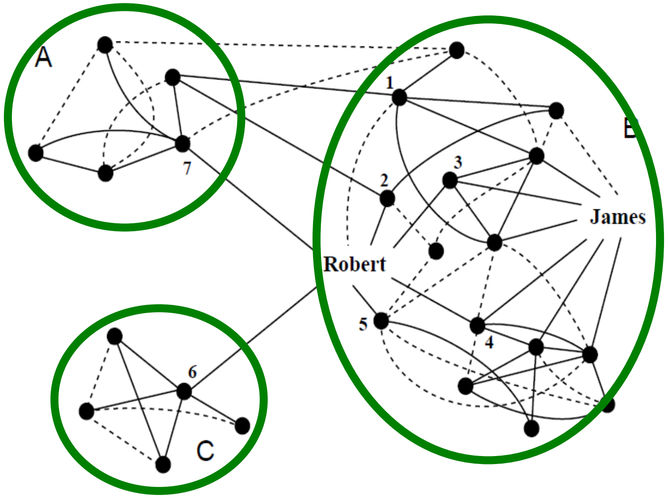
4-degrees of separation [Backstrom-Boldi-Rosa-Ugander-Vigna, 2011]

- Links denote social interactions e.g. friendship
- Identify groups (communities) and group interactions
- Find key influencers in community
- Extract topic interests

Community Detection

- Partition the network into groups such that the inter-group edges are sparse while the intra-group edges are dense.

- Example with 3 communities



- Non-randomness (locality)
  - Relationship tend to cluster
  - If entity P is related to both Q and R, then there is a higher probability than average that Q and R are related

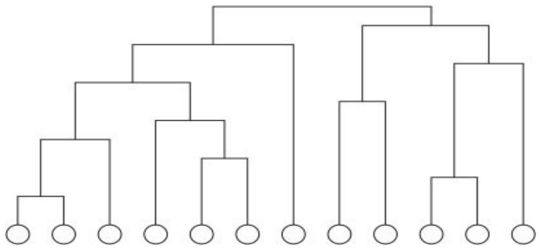
Hierarchical Clustering Approach

- **Define notion of similarity/affinity between nodes**
  - sim(x,y) = # node-disjoint paths between x and y
  - sim(x,y) = # edge-disjoint paths between x and y
  - sim(x,y) = weighted sum of all paths, with longer paths weighted down e.g. Katz

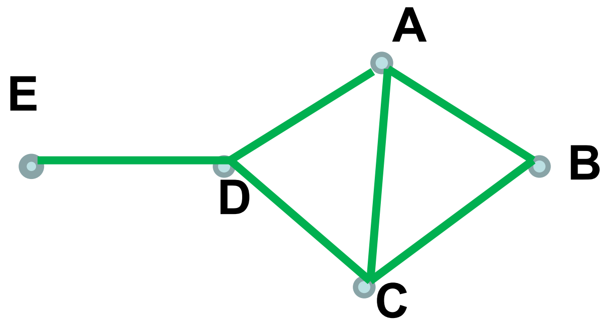
$$\text{Katz}(x,y) = \sum_{k>0} \alpha^k \# \, Paths_k(x, y) \quad where\ 0 < \alpha < 1$$

- **How to compute the node similarities quickly?**

Hierarchical Clustering

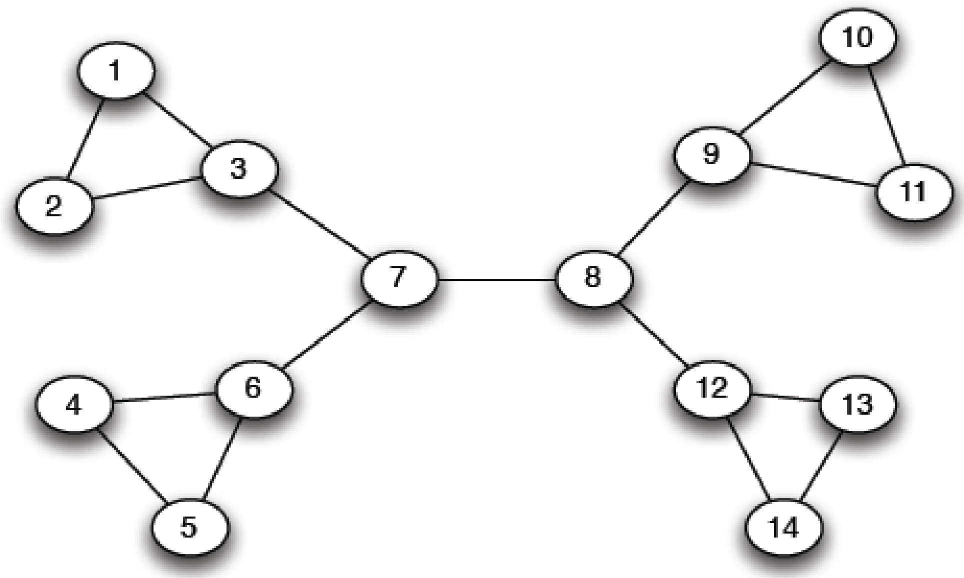


- **Compute all pairwise node similarities for every edge**
- **Repeatedly add edges with largest similarity, leads to a tree or dendrogram**
- **A slice through the dendogram represents a clustering or community structure**
- **Problem: “Misplaces” nodes in the periphery**
  - Which community should node E belong to?



Girvan-Newman Algorithm

- **Hierarchical divisive method**
  - Start with the whole graph
  - Find edges whose removal "partitions" the graph
  - Repeat with each subgraph until single vertices



Which edge to remove?

Edge Importance

- **Need a measure of how important an edge is in keeping the graph connected**
- **Edge betweenness: Number of shortest paths that pass through the edge**
- **Betweenness of an edge (a,b)** ***B(a,b)***
  - For each pair of nodes (x, y), compute number of shortest paths that include (a,b)
  - May have multiple shortest paths between (x,y) ***SP(x,y)***
    - Compute the fraction of those that pass through (a,b)

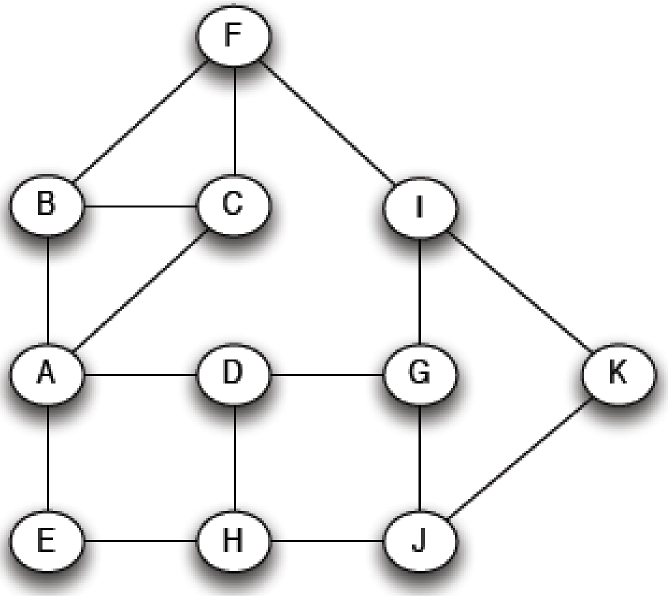
$$B(a,b) = \sum_{x,y \in V} \frac{|SP(x,y) \text{ that include } (a,b)|}{|SP(x,y)|}$$

Edge Betweenness

- An inter-community edge has a higher betweenness compared to an intra-community edge, i.e., more paths between node pairs pass through it.
- Basic algorithm
  - Start with graph G.
  - Repeatedly remove edge with highest betweenness until <some stopping criterion>
  - Communities = resulting components

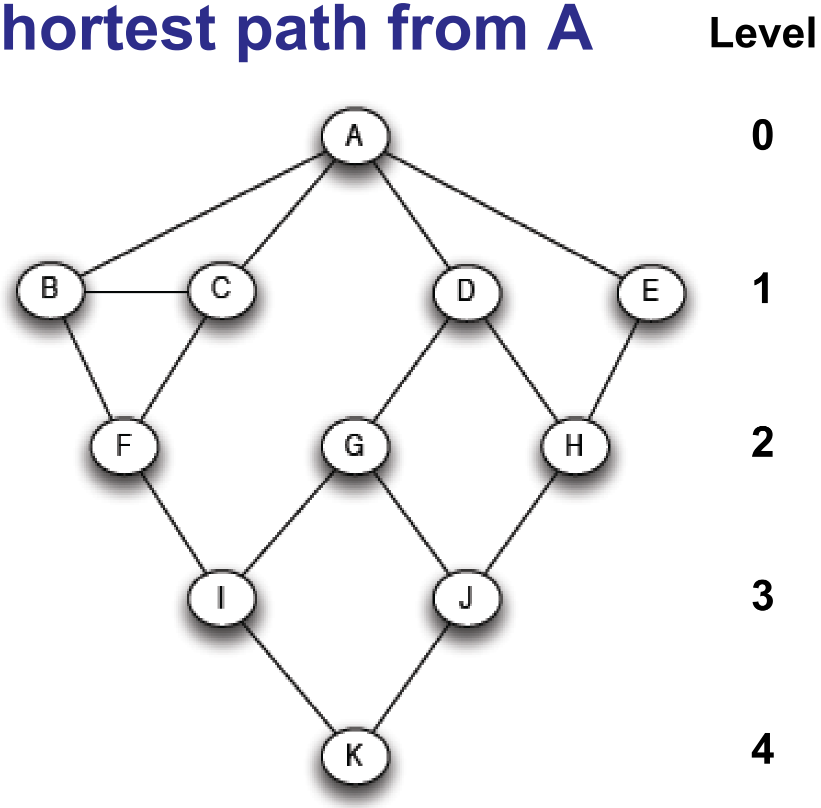
How to Compute Edge Betweenness?

- Want to compute betweenness of paths starting at node A



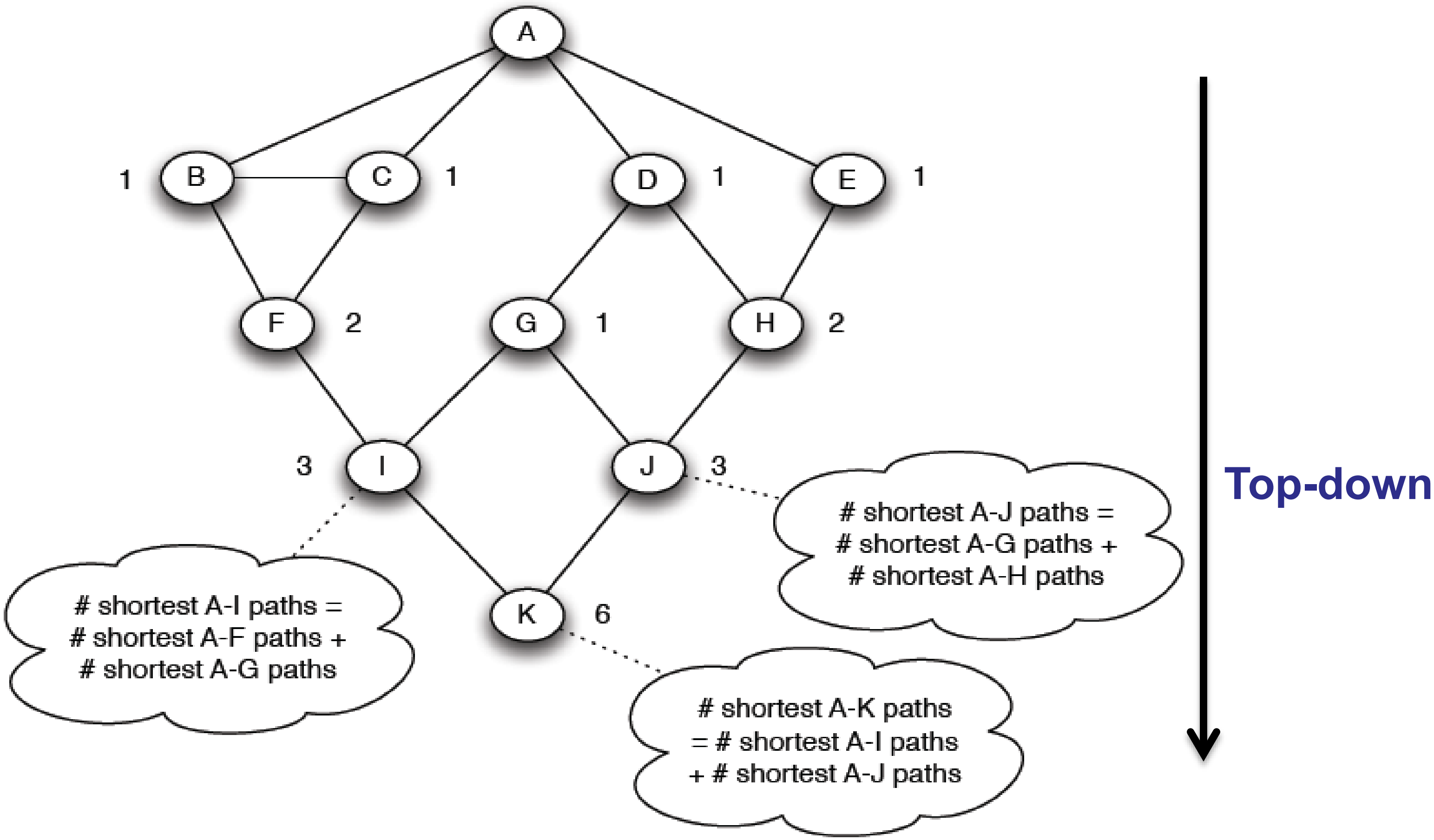
visit the nodes level by level

- Breadth first search starting from A
- Level gives the length of shortest path from A



How to Compute Edge Betweenness?

- Determine the number of shortest paths from A to all other nodes of the network

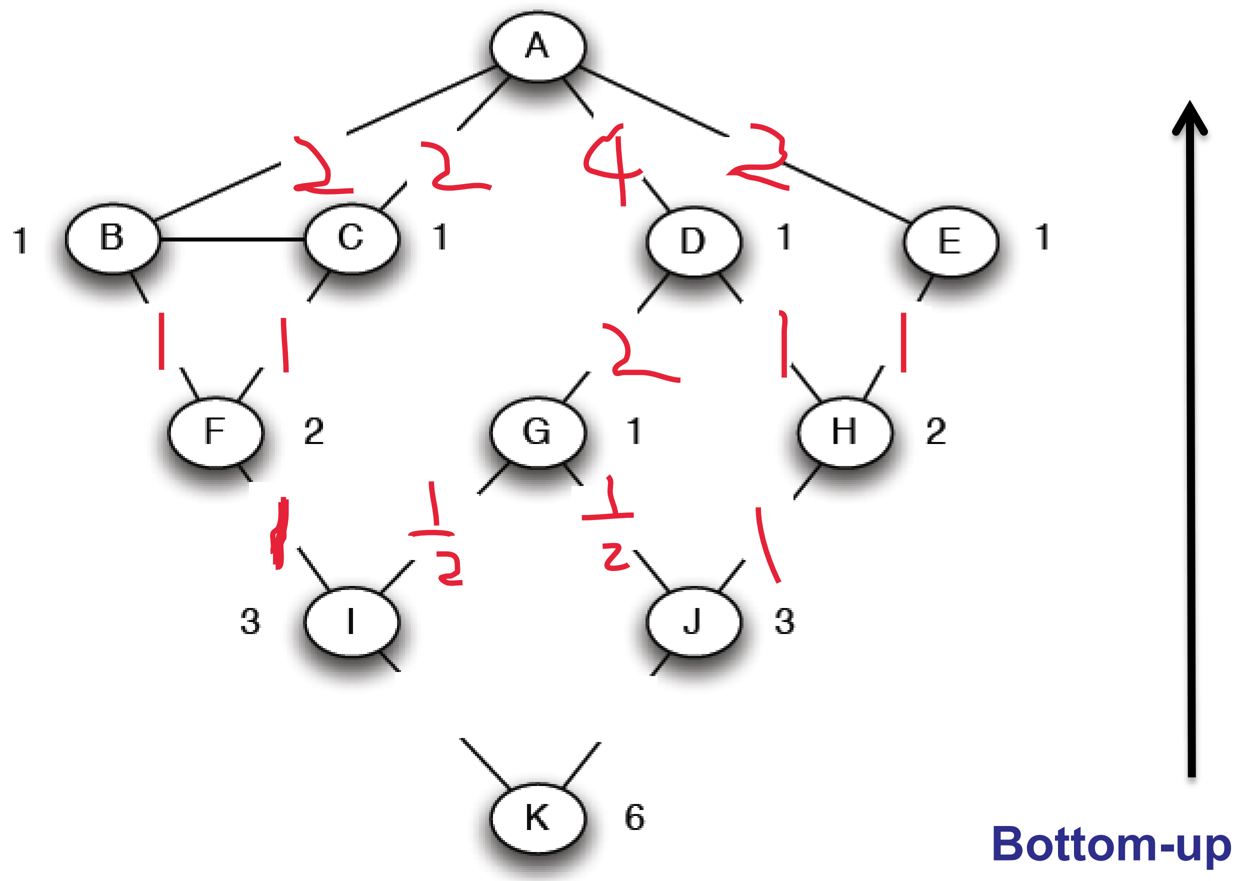


How to Compute Edge Betweenness?

- Compute edge flows by working up the tree
- For every node there is a unit of flow to that node that is divided fractionally to the edges that reach that node

There is a unit of flow to K that reaches K through edges (I,K) and (J,K)

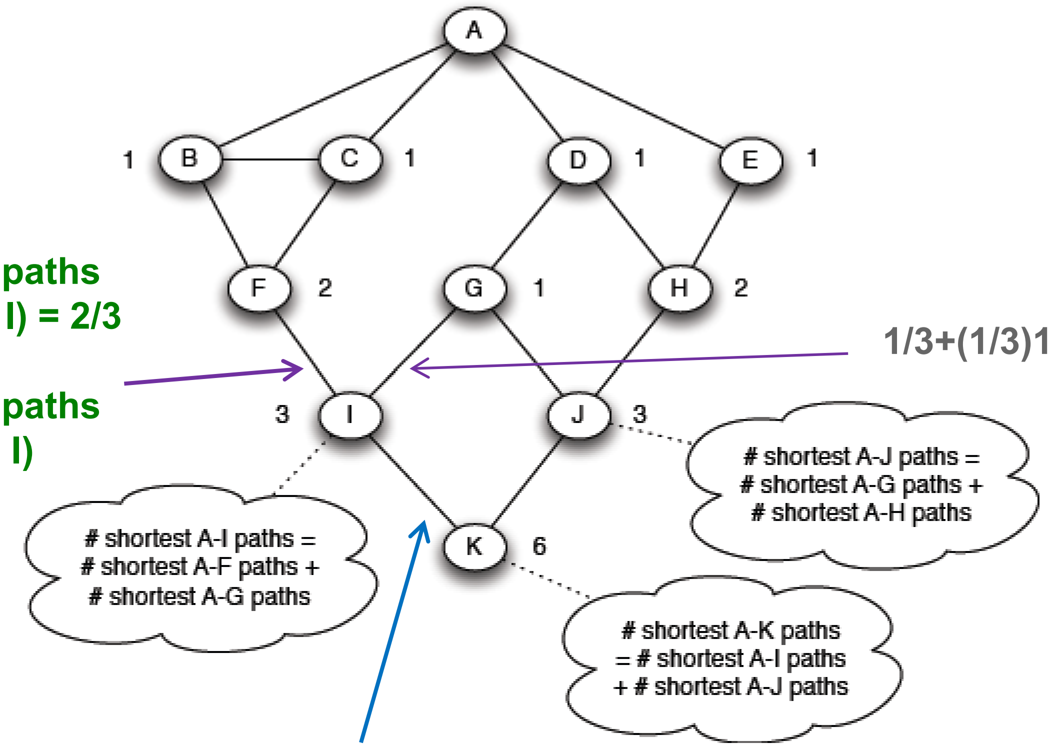
Since there are 3 paths from I to K and 3 paths from J, each edge gets ½ of the flow



How to Compute Edge Betweenness?

- Count the flow through each edge

Portion of the shortest paths to I that go through (F, I) = 2/3
+
Portion of the shortest paths to K that go through (F, I) (1/2)(2/3) = 1/3
= 1



1/3+(1/3)1/2 = 1/2

Portion of the shortest paths to K that go through (I, K) = 3/6 = 1/2

Bottom-up

How to Compute Edge Betweenness?

- Repeat the process for all nodes
- Edge betweenness is given by the sum of the flows computed for each edge

$$B(a,b) = \sum_{x,y \in V} \frac{|SP(x,y)\ that\ include\ (a,b)|}{|SP(x,y)|}$$

Scalability Issues

- m edges and n nodes
- For each target node X, BFS gives the betweenness of every edge w.r.t X in O(m) time
- Final betweenness score for every edge is computed in time O(mn)
- Recalculation takes $O(m^2n)$, not feasible for large networks

Repeat

- *Calculate betweenness of all edges;*
- *Remove edge with highest betweenness, break ties arbitrarily;*

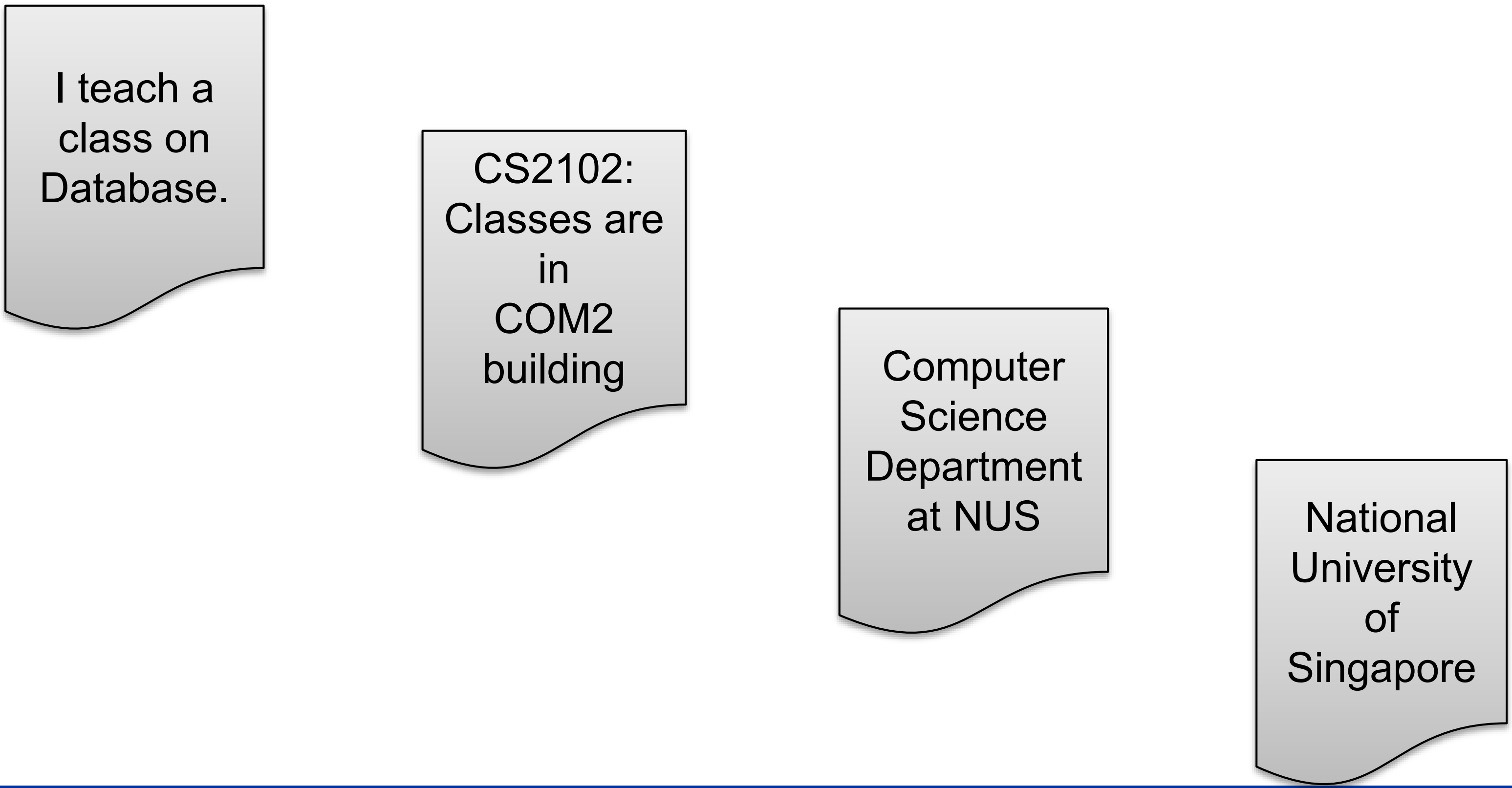
Until <some stopping criterion>

Scaling up CD Algorithm

- Determine which edges need their betweenness recalculated, when an edge is removed.
  - When e is removed, betweenness(e') needs to be recalculated only if e' is in the same connected component as e.
  - Not much pruning if a component is large
- Perhaps it's only important to determine the edge with the next highest betweenness
- Can we maintain enough "state" so that when e is removed, we can recalculate betweenness(e') incrementally?

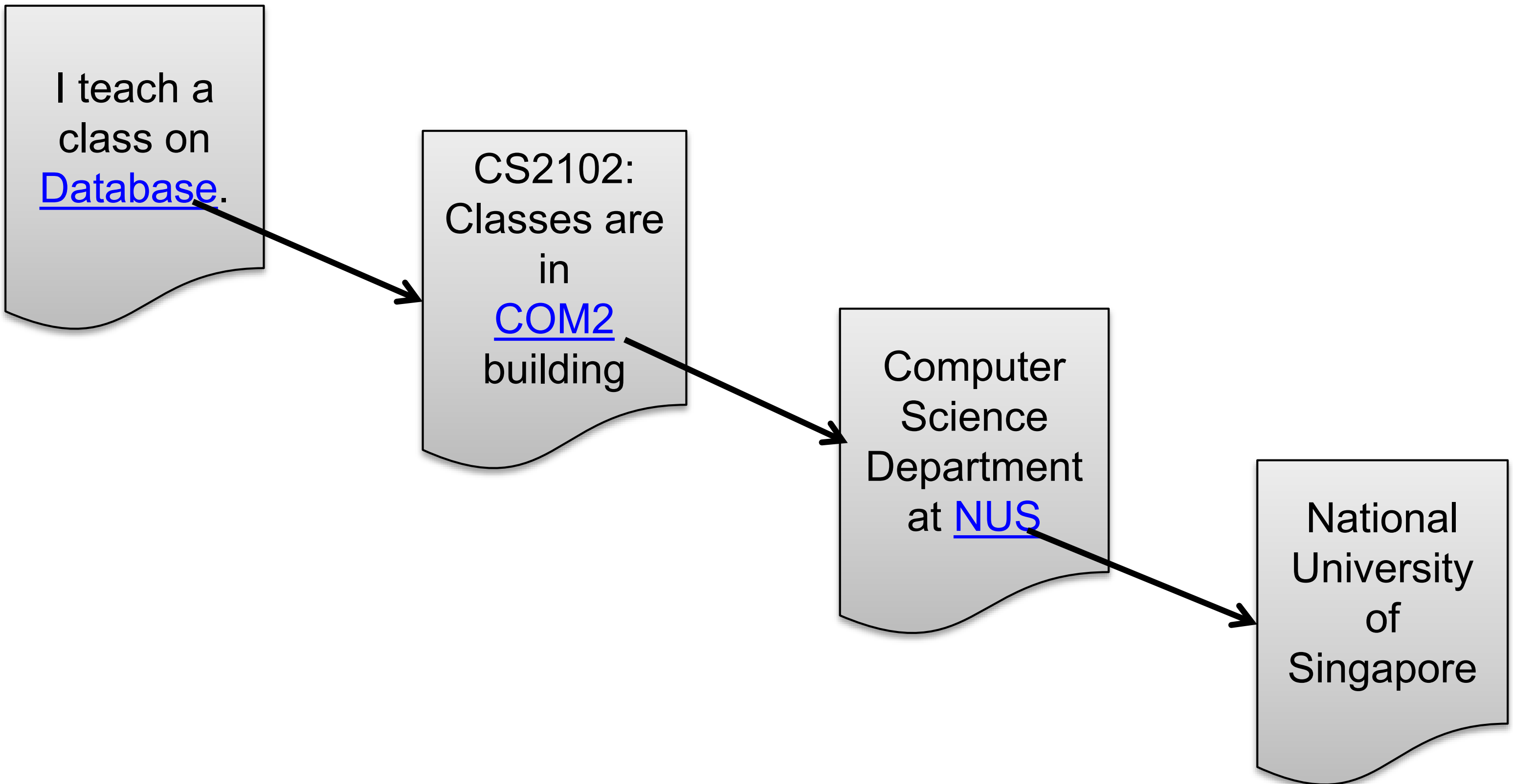
Web as a Graph

- **Nodes: Webpages**
- **Edges: Hyperlinks**

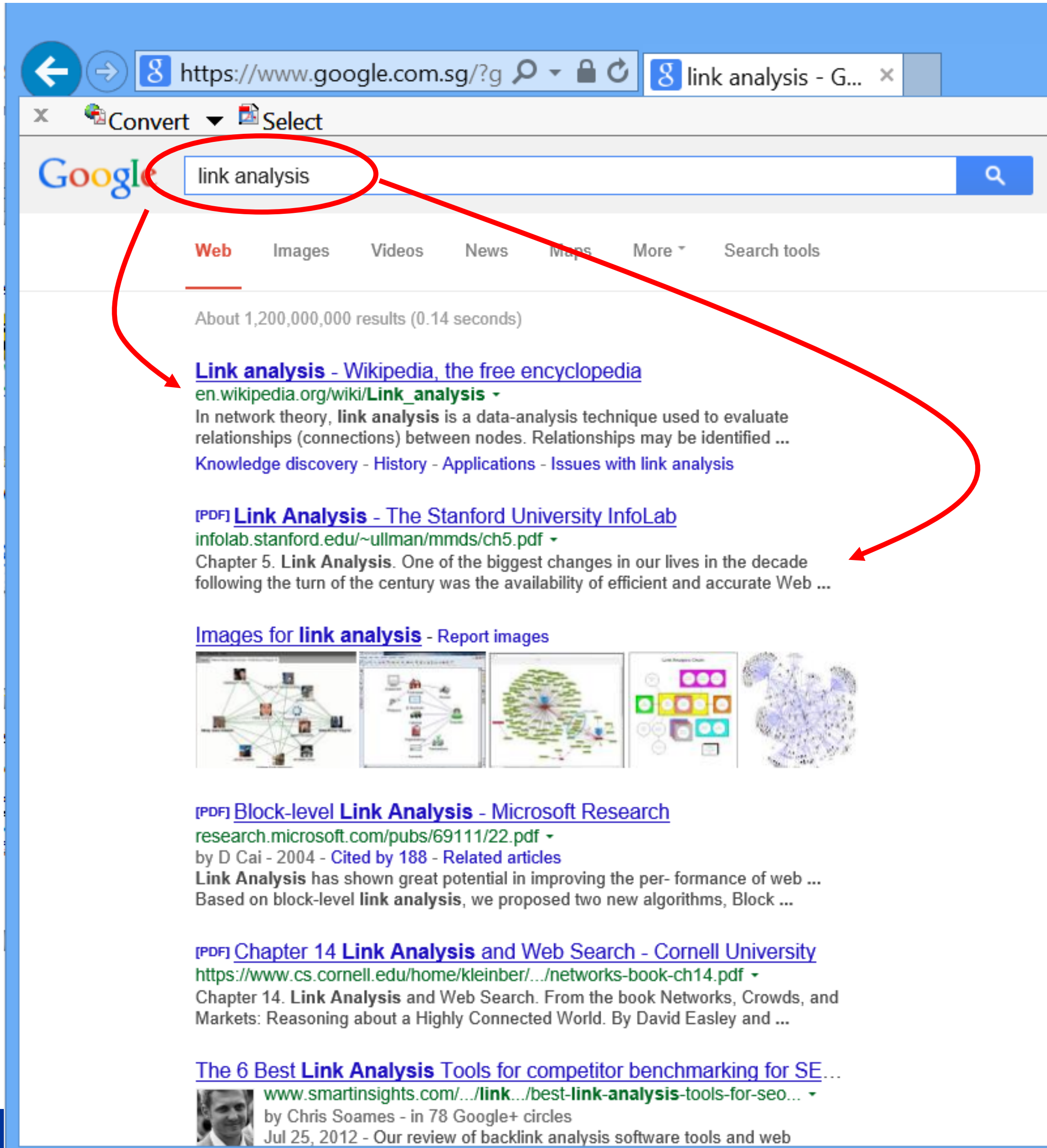


Web as a Graph

- **Nodes: Webpages**
- **Edges: Hyperlinks**



Web Search



- How does the search engine decide which page should be ranked higher?

Web Search - Challenges

- Web contains many sources of information.
  - Who to “trust”?

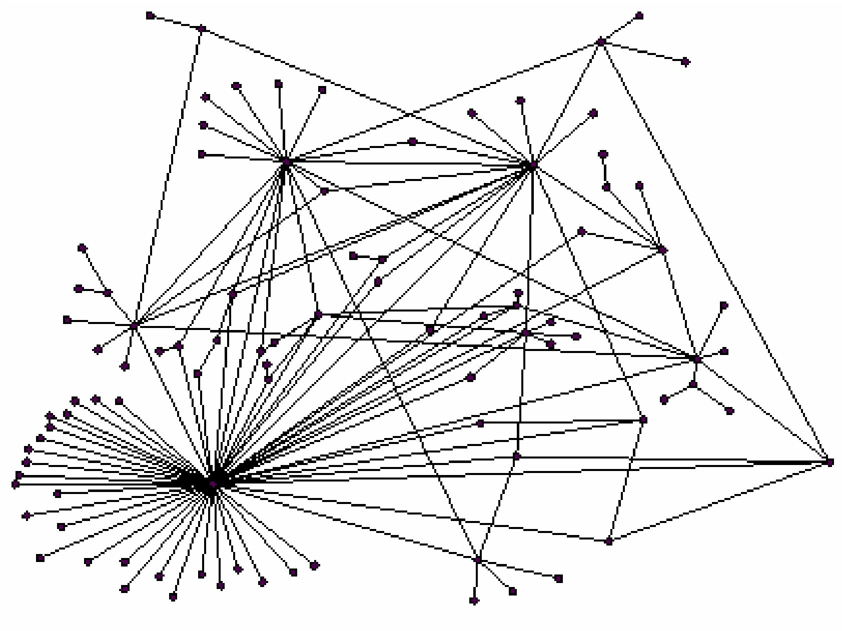
- What is the “best” answer to query “newspaper”?
  - No single right answer

Link Analysis

- **The Web is not just a collection of documents**
  - The hyperlinks are important
- **A link from page *A* to page *B* may indicate**
  - *A* is related to *B*, or
  - *A* is recommending, citing, voting for, or endorsing *B*
- **Types of links:**
  - Referential – *click here and get back home*
  - Informational – *click here to get more detail*
- **Links influence the ranking of web pages and thus have commercial value**

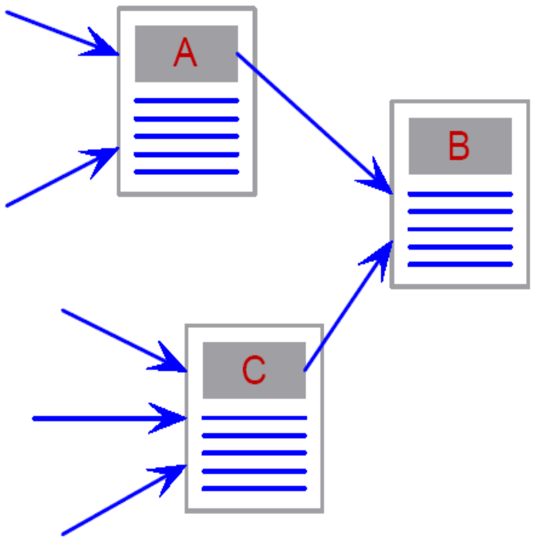
Importance of Web Pages

- Not all web pages are equally important
- A page is important if it is pointed to by other important pages (recursion)



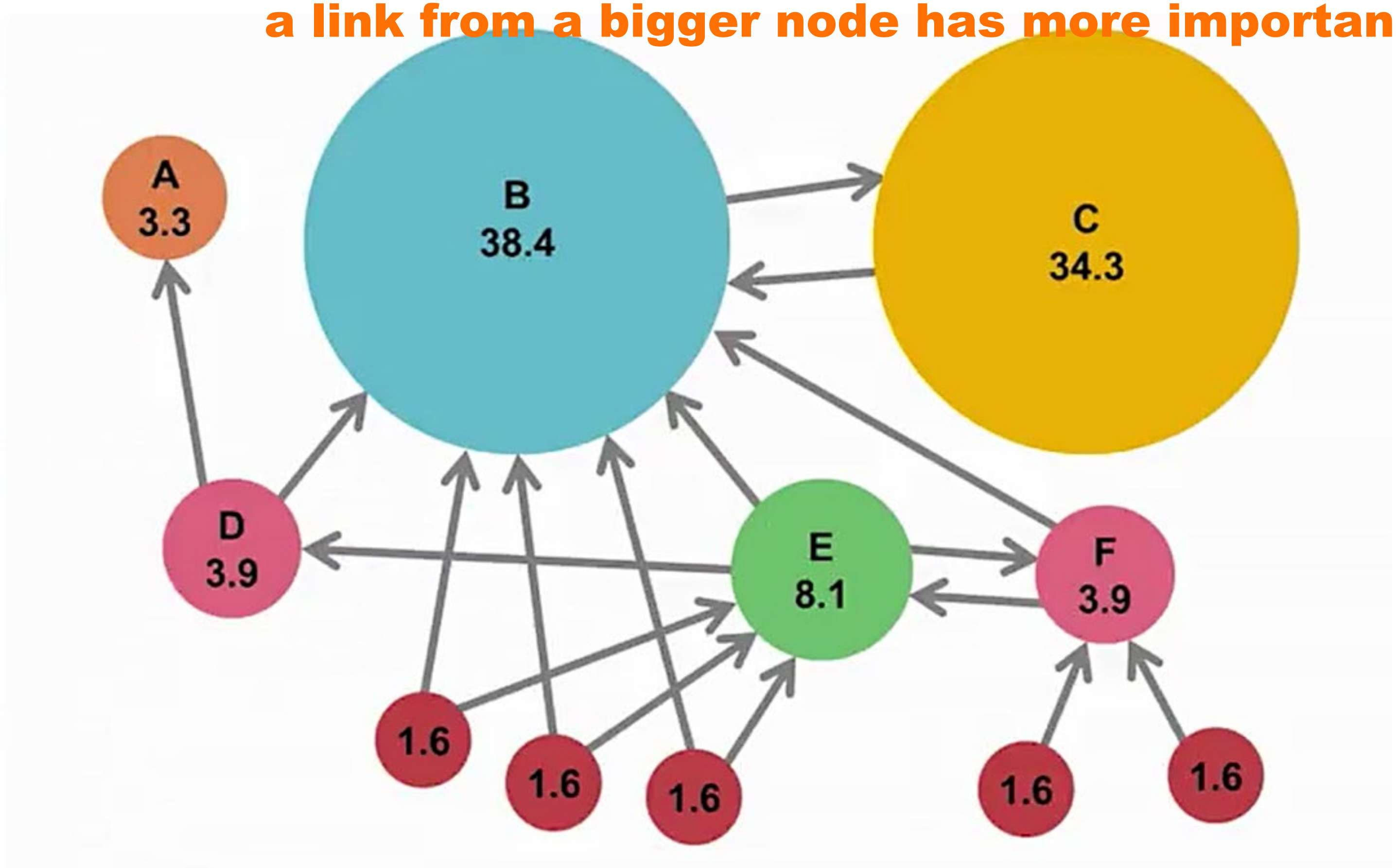
PageRank

- **Idea: Links as votes**
  - A page is more important if it has more links
- **Incoming links to a page is a measure of importance and authority of the page**
  - www.stanford.edu has 23,400 in-links
  - www.joe-schmoe.com has 1 in-link
- **Are all incoming links equal?**
  - Links from important pages count more!



Example PageRank Scores

a link from a bigger node has more importance

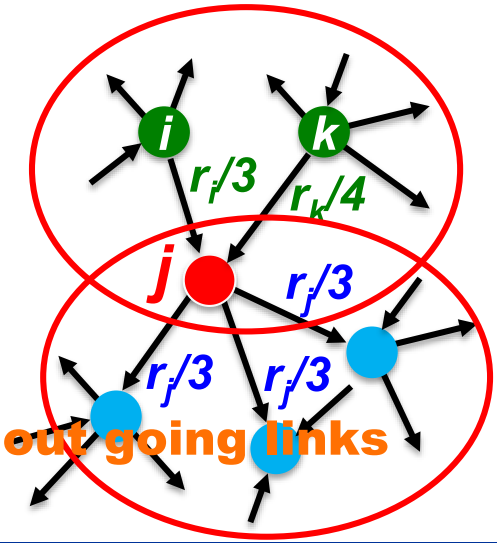


- A “vote” from an important page is worth more
- A page is important if it is pointed to by other important pages

Recursive Formulation

- A link's vote is proportional to the importance of its source page
- If page j with importance r_j has n out-links, each link gets r_j / n votes
- Page j's own importance is the sum of the votes on its in-links

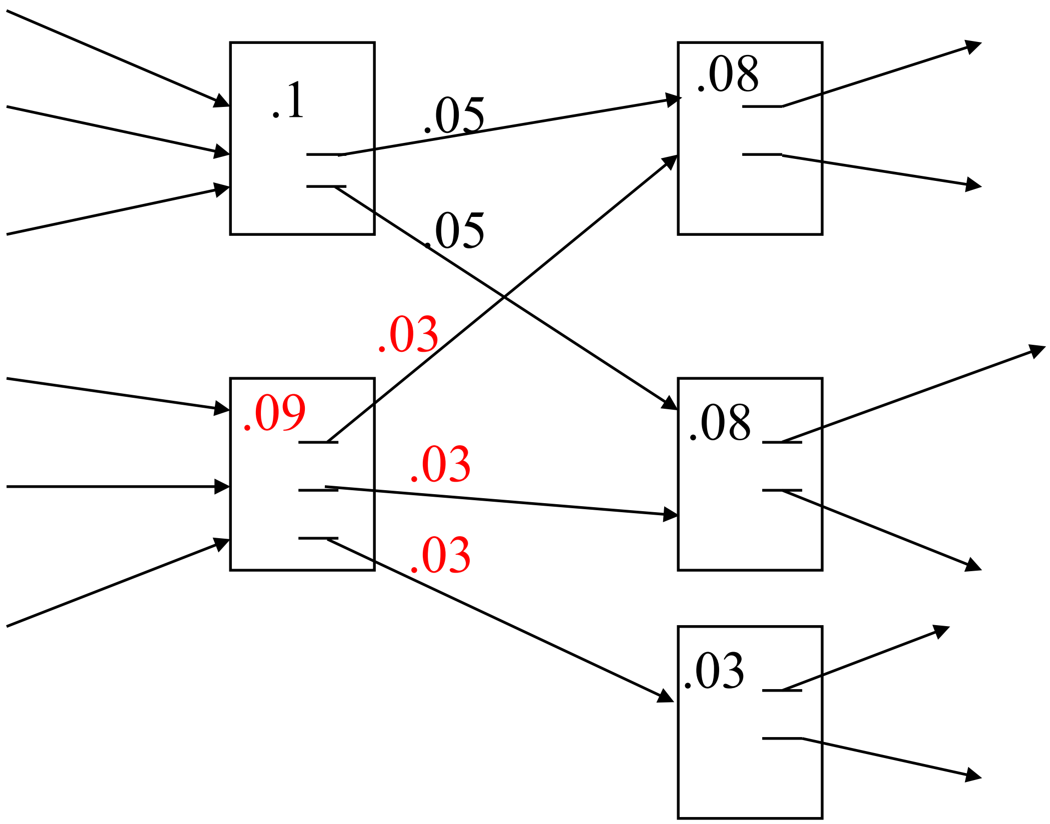
$$r_j = r_i/3 + r_k/4$$



how many out going links

Flow Model

- Can view it as a process of PageRank “flowing” from pages to the pages they point to

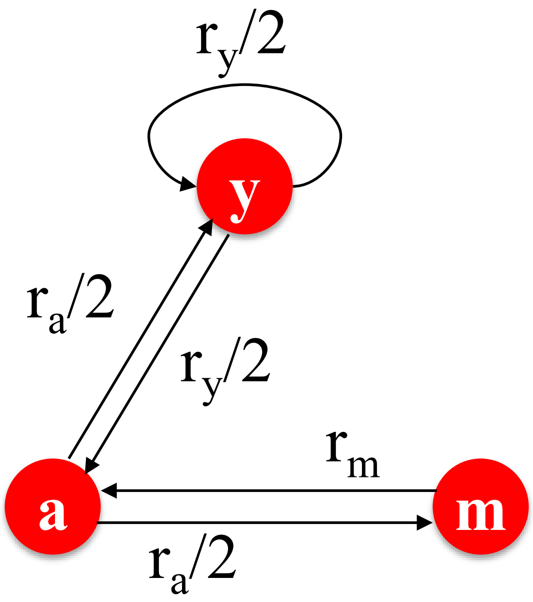


Flow Model

- **Define a rank r_j for page j**

$$r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d_i}}$$

d_i **is the out-degree of node** i



- **Flow Equations**

$r_y = r_y/2 + r_a/2$

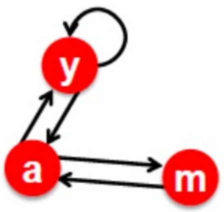
$r_a = r_y/2 + r_m$

$r_m = r_a/2$

the incoming edges

Matrix Formulation

- **Stochastic adjacency matrix *M***
  - Let page i has d_i outlinks
  - If $i \rightarrow j$, then $M_{ji} = 1/d_i$ else $M_{ji} = 0$
  - ***M*** is a column stochastic matrix
    - **Columns sum to 1**



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

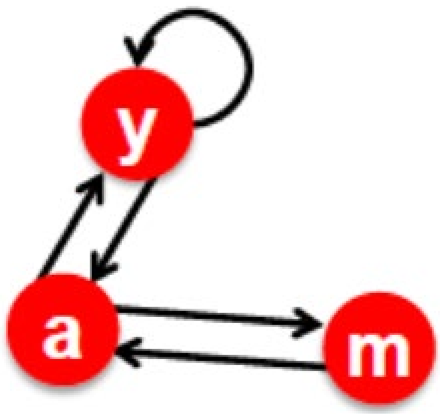
- **Rank vector *r***
  - Vector with one entry per page
  - r_i is the importance score of page i

$$\sum_i r_i = 1$$

- **Flow equations can be written in matrix form $r = M \cdot r$**

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

Example Flow Equations and M



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$
\begin{aligned}
r_y &= r_y/2 + r_a/2 \\
r_a &= r_y/2 + r_m \\
r_m &= r_a/2
\end{aligned}
\quad \equiv \quad
\begin{bmatrix} y \\ a \\ m \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} y \\ a \\ m \end{bmatrix}
$$

Power Iteration Method

- **Given a web graph with *n* nodes, where the nodes are pages and edges are hyperlinks**
- **Power iteration – simple iterative scheme**
  - Suppose there are N web pages
  - Initialize $\mathbf{r}^{(0)} = [\ 1/N, \ldots., 1/N\ ]^T$
  - Iterate: $\mathbf{r}^{(t+1)} = \mathbf{M} \cdot \mathbf{r}^{(t)}$
- **Stop when $|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}|_1 < \varepsilon$**
  - $|\mathbf{x}|_1 = \sum_{i\in[1,N]} |x_i|$ is the L_1 norm
  - Can use any other vector norm e.g., Euclidean

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_i}$$

d_i …. out-degree of node i

Example

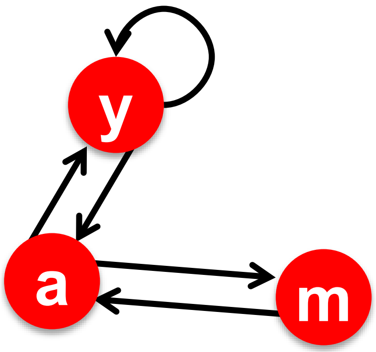
- **Power Iteration:**

  Set $r_j = 1/N$

  1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$

  2: $r = r'$

  Goto 1



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

- **Example:**

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & & & & & \\ 1/3 & & & & & \end{matrix}$$

Iteration 0, 1, 2, …

PageRank

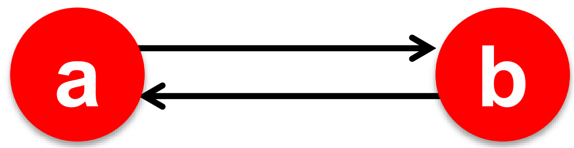
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d_i}}$$

or equivalently

$$r = Mr$$

- **Questions:**
  1. Does this converge?
  2. Does it converge to what we want?

Does this converge?

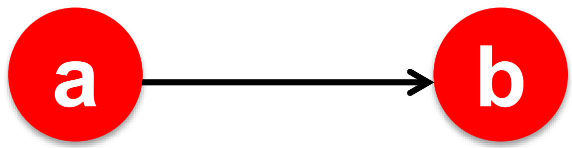


$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_\mathrm{i}}$$

$$\begin{array}{c} r_a \\ r_b \end{array} = \begin{array}{cccc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{array}$$

Iteration 0, 1, 2, …

Does it converge to what we want?



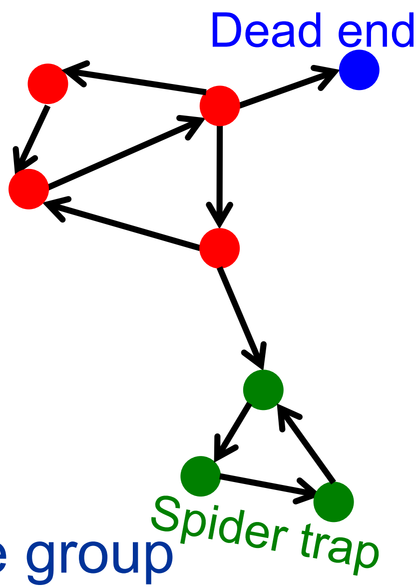
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_\mathrm{i}}$$

$$\begin{array}{l} \mathbf{r_a} \\ \mathbf{r_b} \end{array} = \begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{array}$$

Iteration 0, 1, 2, …

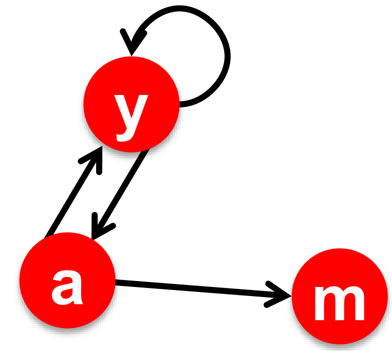
Problems on Real Web

- **Imagine a random web surfer**
  - At any time *t*, surfer is on some page *i*
  - At time *t+1*, surfer follows an out-link from *i* uniformly at random
  - Ends up on some page *j* linked from *i*
- **Dead ends**
  - A page has no out-links
  - Random walk has “nowhere” to go to
  - Such pages cause importance to “leak out”
- **Spider traps**
  - A group of pages have no out-links out of the group
  - Random walk gets “stuck” in a trap
  - Eventually spider traps absorb all importance



Problem: Dead Ends

- A page with no out-links
- Random walk has "nowhere" to go to
- All importance "leaks out of" the Web!
- Matrix is not stochastic so initial assumptions are not met



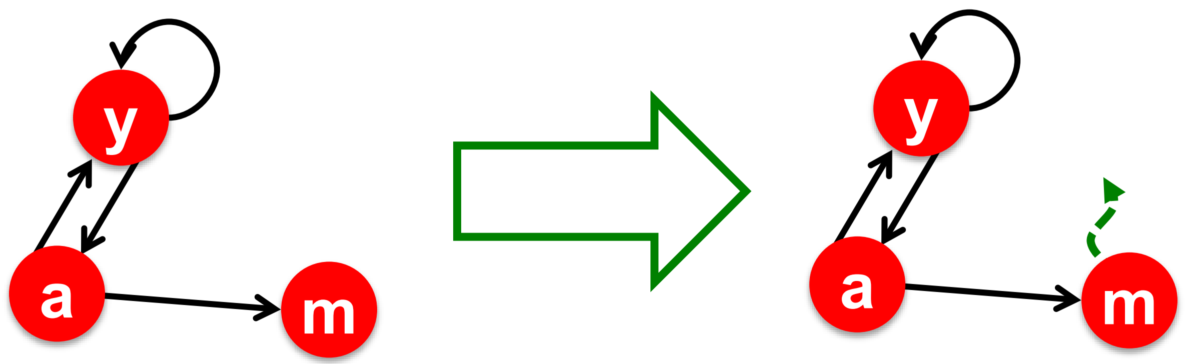
	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/6 \\ 1/6 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/6 \\ 1/12 \end{bmatrix} \begin{bmatrix} 5/24 \\ 1/8 \\ 1/12 \end{bmatrix} \cdots \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Iteration 0, 1, 2, …

Solution: Teleport

- Adjust the matrix to allow a surfer to jump to some random page from dead ends



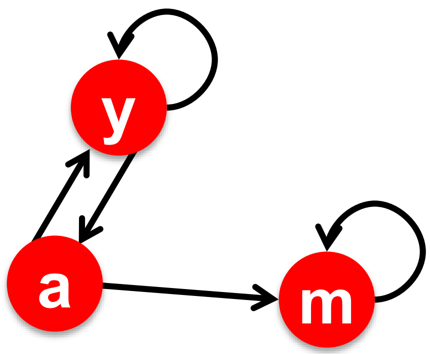
	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

	y	a	m
y	½	½	⅓
a	½	0	⅓
m	0	½	⅓

Problem: Spider Traps

- A group of pages with no links out of the group
- Random walk gets "stuck" in a trap
- Accumulate all the importance of the Web

All the PageRank score gets "trapped" in node m

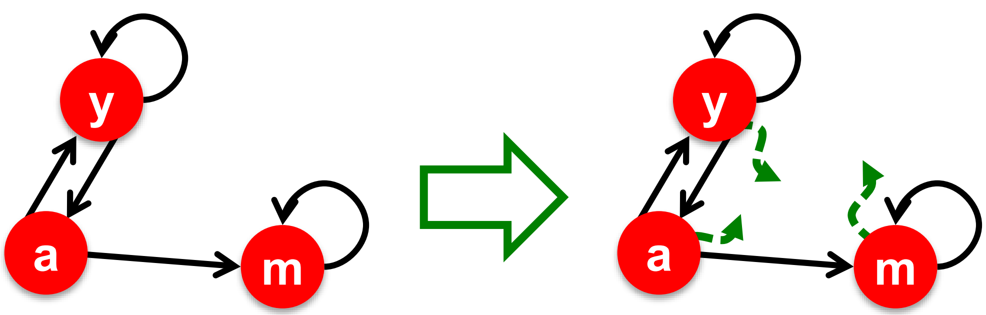


	y	a	m
y	½	½	0
a	½	0	0
m	0	½	1

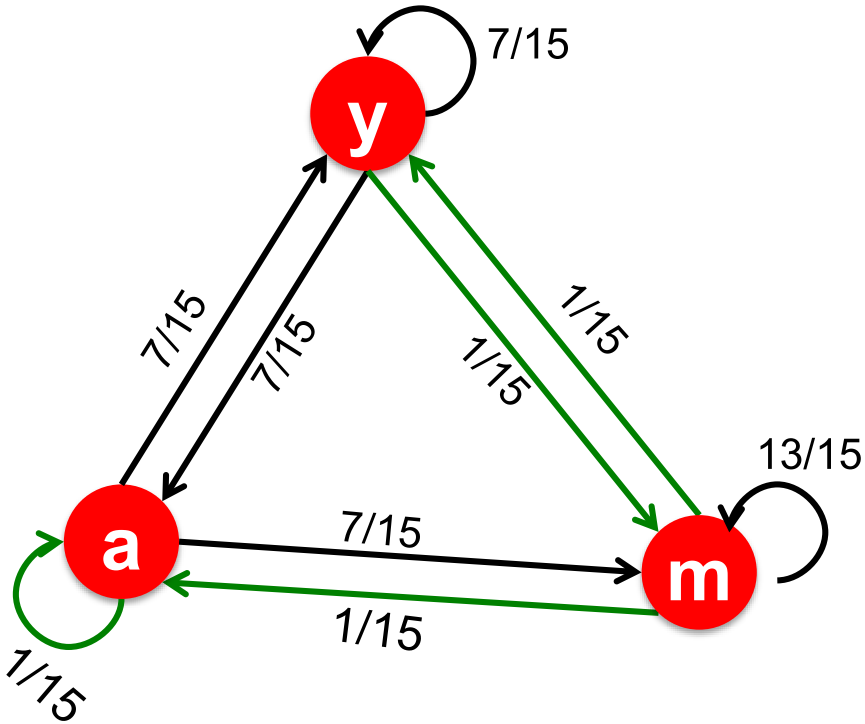
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/6 \\ 1/2 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/6 \\ 7/12 \end{bmatrix} \begin{bmatrix} 5/24 \\ 1/8 \\ 2/3 \end{bmatrix} \cdots\cdots \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Solution: Teleport

- **At each time step, a random surfer has two options**
  - With probability β, follow a link at random
  - With probability **1-**β, jump to some random page
  - Common values for β are in the range 0.8 to 0.9
- **Surfer will teleport out of spider trap within a few time steps**



Random Teleports ($\beta = 0.8$)



$$0.8 \overset{\textbf{M}}{\boxed{\begin{matrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{matrix}}} + 0.2 \overset{[1/N]_{NxN}}{\boxed{\begin{matrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{matrix}}}$$

$$\begin{array}{c} y \\ a \\ m \end{array} \underset{\textbf{A}}{\boxed{\begin{matrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{matrix}}}$$

$$\begin{matrix} \mathbf{r}_y \\ \mathbf{r}_a \\ \mathbf{r}_m \end{matrix} = \begin{matrix} 1/3 \\ 1/3 \\ 1/3 \end{matrix} \quad \begin{matrix} 0.33 \\ 0.20 \\ 0.46 \end{matrix} \quad \begin{matrix} 0.24 \\ 0.20 \\ 0.52 \end{matrix} \quad \begin{matrix} 0.26 \\ 0.18 \\ 0.56 \end{matrix} \quad \dots \quad \begin{matrix} 7/33 \\ 5/33 \\ 21/33 \end{matrix}$$

Limitations of PageRank

- **Measures generic popularity of a page**
  - Ignore or miss topic-specific authorities
  - **Solution:** Topic-specific PageRank
- **Uses a single measure of importance**
  - Other models of importance
  - **Solution:** Hubs-and-Authorities
- **Susceptible to link spam**
  - Artificial link topologies created in order to boost page rank
  - **Solution:** TrustRank

Topic-Specific PageRank

- **Instead of generic popularity, can we measure popularity within a topic?**
- **Goal: Evaluate Web pages not just according to their popularity, but by how close they are to a particular topic, e.g. “sports” or “history”**
- **Allows search queries to be answered based on interests of the user**
  - **Example:** Is “Jaguar” an animal, the automobile, or a version of MAC OS?

Topic-Specific PageRank

- **Recall random walker has a small probability of teleporting at any step**
  - Standard PageRank: Any page with equal probability
  - Topic Specific PageRank: Teleport set is restricted to a topic-specific set of “relevant” pages
- **Idea: Bias the random walk**
  - When random walker teleports, pick a page from a set $\boldsymbol{S}$ of web pages
  - $\boldsymbol{S}$ contains only pages that are relevant to the topic
  - Get a different rank vector $\boldsymbol{r_S}$ for each teleport set $\boldsymbol{S}$

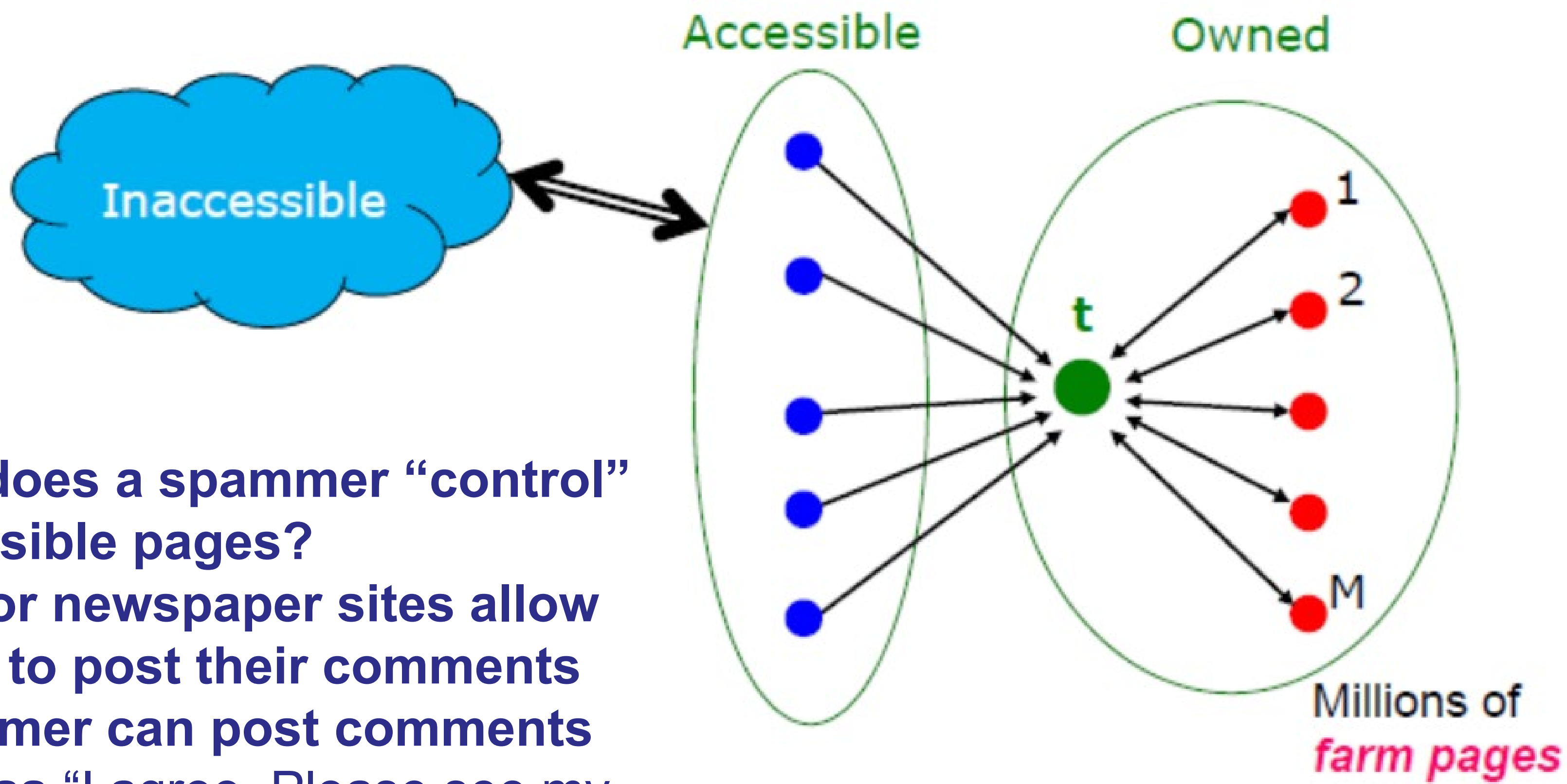
Topic-Specific PageRank

- **Decide on topics to create PageRank vectors**
  - Open Directory (DMOZ) (www.dmoz.org)
  - The 16 DMOZ top-level categories: arts, business, sports, …
- **Pick a teleport set for each of these topics, and compute the topic-sensitive PageRank vector for that topic**
- **Determine the topic that is most relevant for a query**
  - User picks from a menu
  - Query context e.g., query from a web page on a known topic
  - User context e.g., user's bookmarks
- **Use the PageRank vectors for that topic to order results to the search query**

TrustRank – Combating Web Spam

- **Spamming**
  - Any deliberate action to boost a web page's position in search engine results, incommensurate with page's real value
  - Approximately **10-15%** of web pages are spam
- **Link Spam**
  - Create link structures that boost the PageRank of a particular page

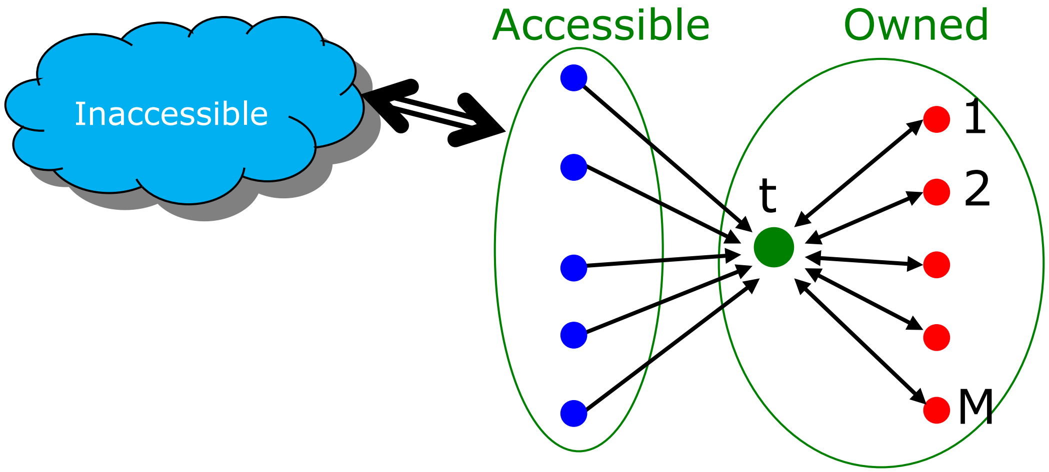
Spammer’s View of the Web



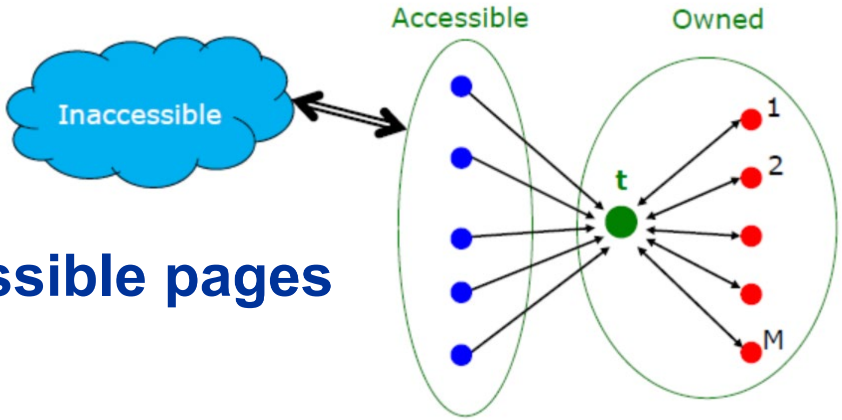
- **How does a spammer “control” accessible pages?**
- **Blog or newspaper sites allow users to post their comments**
- **Spammer can post comments such as** “I agree. Please see my article at www.mySpamFarm.com”

Link Farms

- Spammer's goal is to maximize the PageRank of target page *t*
- Get as many links from accessible pages as possible to target page *t*
- Construct "link farm" to get PageRank multiplier effect



Analysis



x: PageRank contributed by accessible pages

y: PageRank of target page t

N: Total number of web pages

Rank of each "farm" page $= \frac{\beta y}{M} + \frac{1-\beta}{N}$

$$y = x + \beta M \left[\frac{\beta y}{M} + \frac{1-\beta}{N}\right] + \frac{1-\beta}{N}$$

$$y = \frac{x}{1-\beta^2} + c\frac{M}{N} \quad \text{where } c = \frac{\beta}{1+\beta}$$

Let β = 0.85. Then $1/(1 - \beta^2)$ = 3.6, and c = 0.46

- External PageRank (x) increased by 360%!
- Obtain additional amount of PageRank that is 46% of the fraction of the Web, M/N, that is in the spam farm
- By making M large, we can make y as large as we want

How to Combat Link Spam?

- **Detect and blacklist structures that look like spam farms**
  - One page links to a very large number of pages, each of which links back to it
  - Leads to more sophisticated way of hiding spam farms, and detecting them…
- **TrustRank: Topic-specific PageRank with a teleport set of trusted pages**
  - e.g, **.edu** domains, **.gov** domains, etc
  - Lower the score of spam pages

TrustRank

- **Basic principle: Approximate isolation**
  - It is rare for a “good” (trustworthy) page to point to a “bad” (spam) page
- **Sample a set of seed pages from the web**
- **An oracle (human) identifies the good pages and the spam pages in the seed set**
  - Expensive task, so keep seed set small
  - Subset of pages in the seed set that are identified as good are called the **trusted pages**

Trust Propagation

- **Perform a topic-sensitive PageRank with the trusted pages as the teleport set**
- **Propagate trust through links**
  - Each page gets a trust value between 0 and 1
- **Use a threshold value and mark all pages below the trust threshold as spam**

Trust Propagation (Simple Model)

- **Set trust of each trusted page to 1**
- **Suppose trust of page p is t_p**
  - p has a set of out-links o_p
- **For each $q \in o_p$, p confers the trust to q**
  - $\beta\, t_p / |o_p|$ for $0 < \beta < 1$
- **Trust is additive**
  - Trust of p is the sum of the trust conferred on p by all its in-linked pages
- **Trust attenuation**
  - Degree of trust conferred by a trusted page decreases with the distance in the graph
- **Trust splitting**
  - The larger the number of out-links, the less scrutiny the page author gives each out-link; trust is split across out-links

Picking the Seed Set

- **Two conflicting considerations:**
  - Human has to inspect each seed page → seed set must be small
  - Must ensure every **good page** gets adequate trust rank → need to make all good pages reachable from seed set by short paths

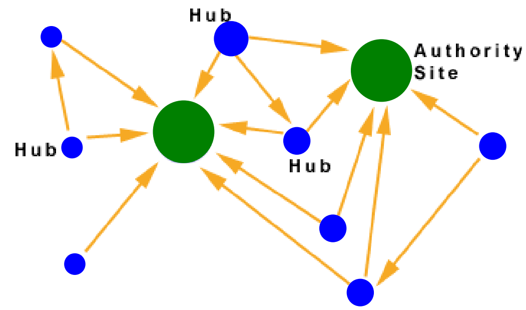
1. Use PageRank to pick the top-k pages

- Theory is a bad page cannot have very high rank

2. Use trusted domains with controlled membership

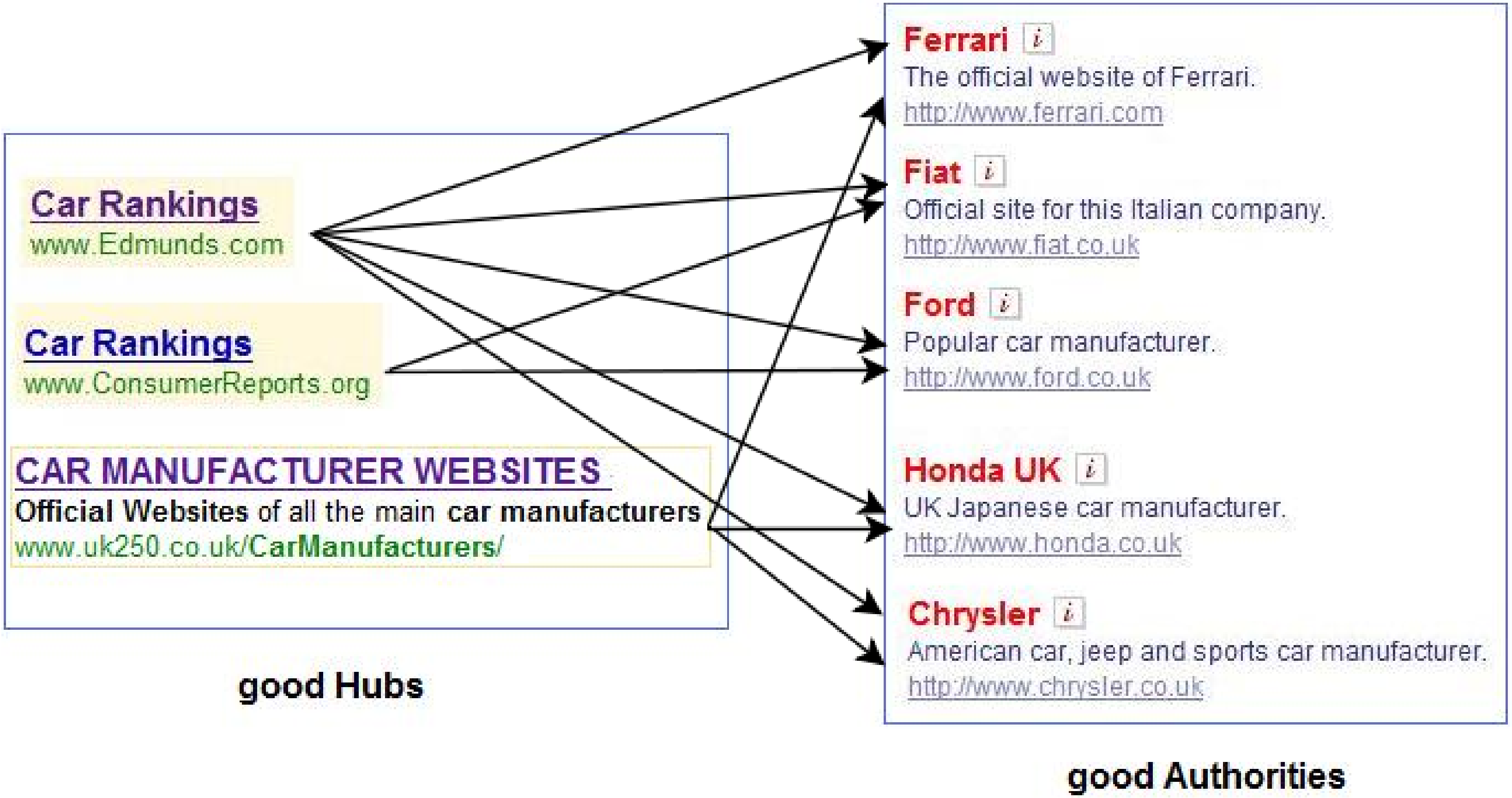
- E.g. university pages (.edu)
  or government pages (.gov)

Hubs and Authorities



- **HITS (Hypertext-Induced Topic Selection)**
  - Originally developed to rank answer pages to a query, same logic can be applied to rank Web pages
  - A measure of importance of pages (similar to PageRank)
- **Goal: Don't just find pages with relevant content, find pages that link to these relevant content**
  - E.g., don't just find trusted news, find also "experts" that point you to the quality news
- **Idea: Links are votes; A page is more important if it has more links**
    - In-coming links? Out-going links?

Hubs and Authorities



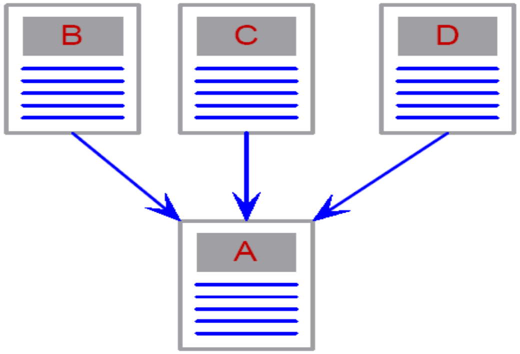
Query: Top automobile makers

Hub and Authorities

- **Each page has 2 roles, hence 2 scores**

(1) Authority

- Pages that contain useful information (quality in providing content) e.g., course home pages
- Voted or linked by "experts" (hubs)
- A good authority is linked from many good hubs

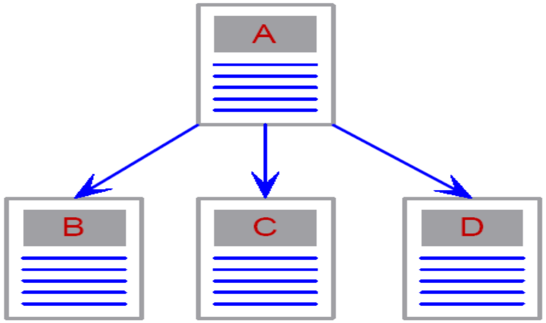


A is an authority

(2) Hub

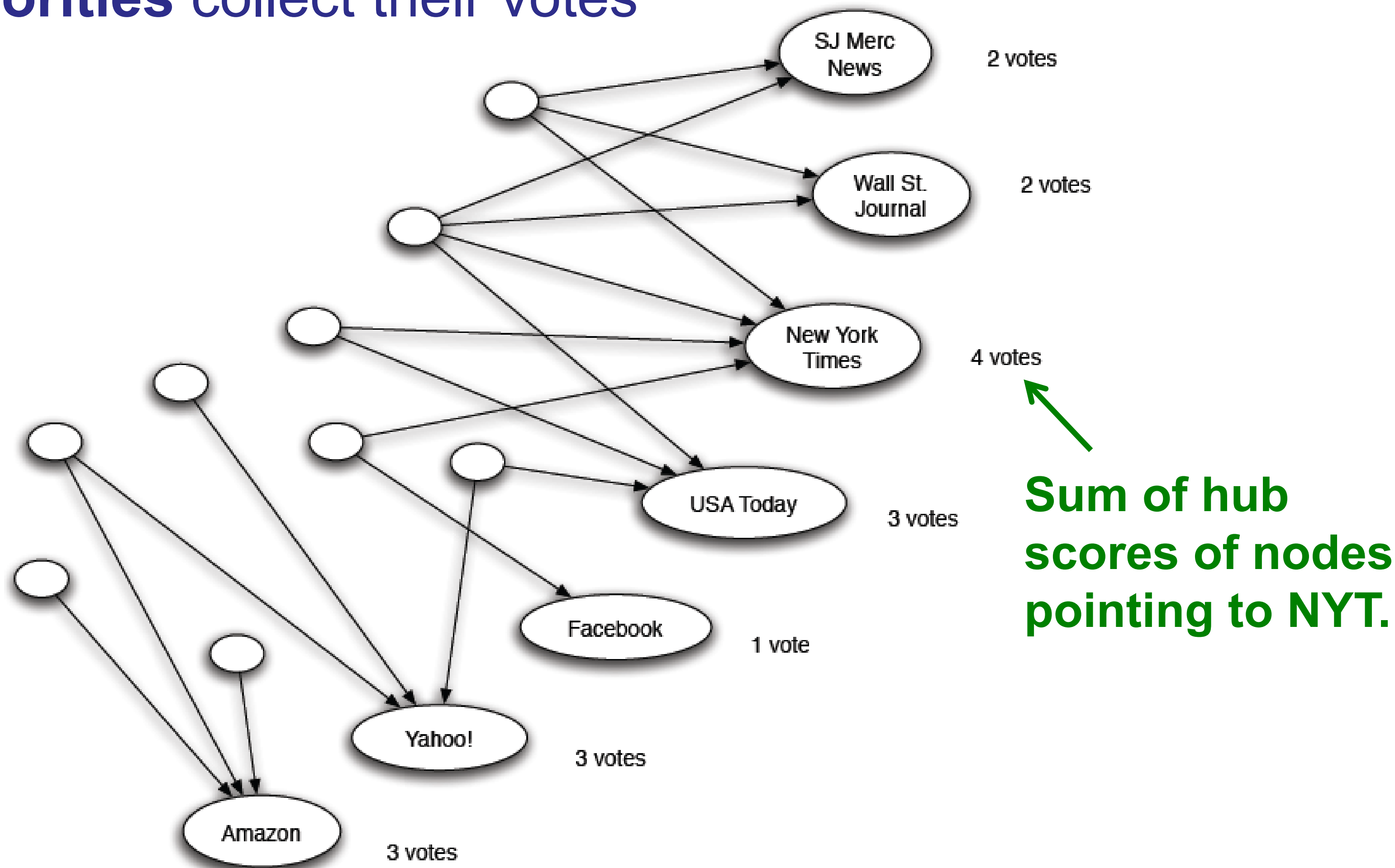
- Pages that link to authorities (quality as an expert) e.g., list of course home pages
- Votes of authorities pointed to
- A good hub links to many authorities

A is a hub



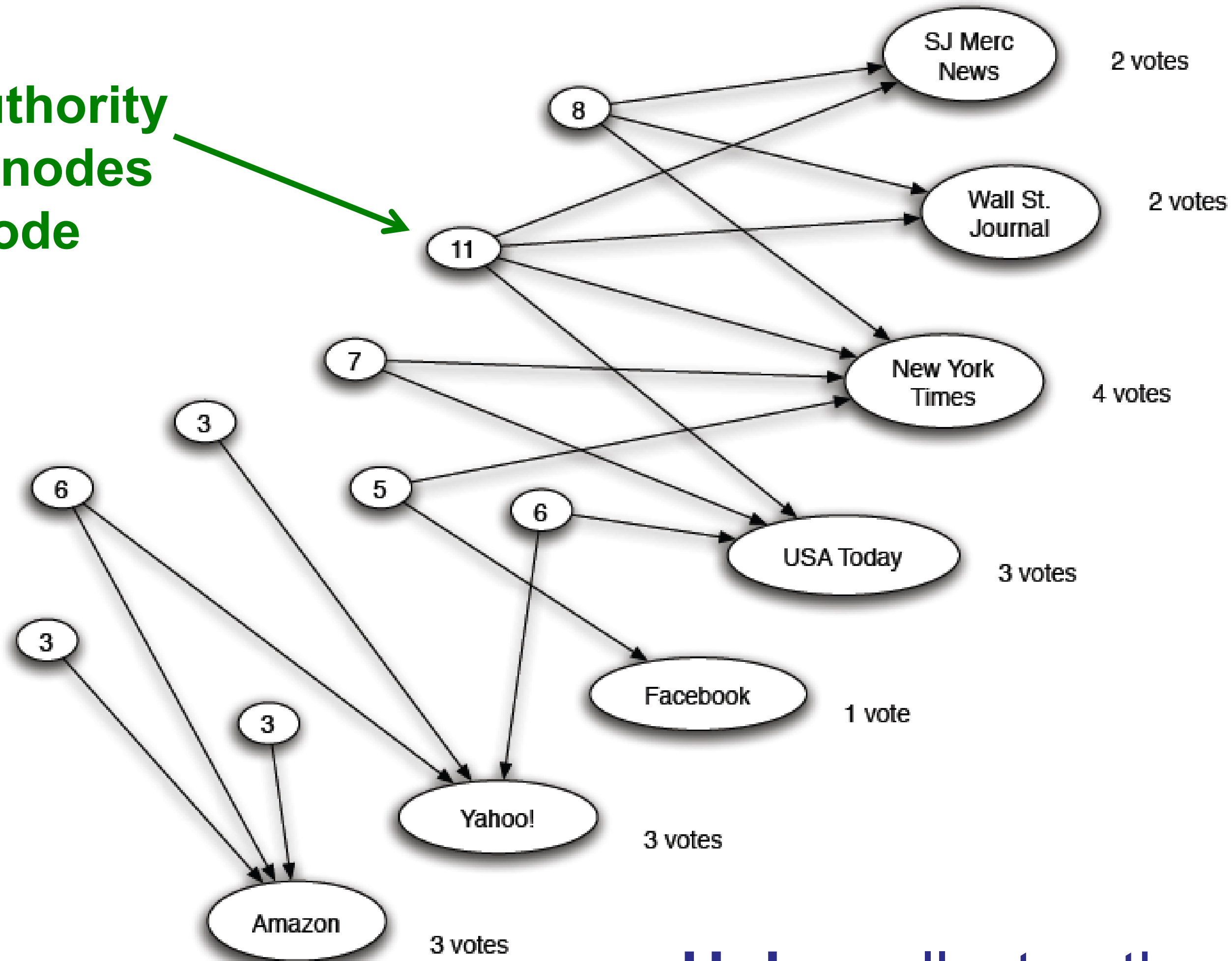
Counting in-links: Authority

- Each page starts with **hub** score 1
- **Authorities** collect their votes



Expert Quality: Hub

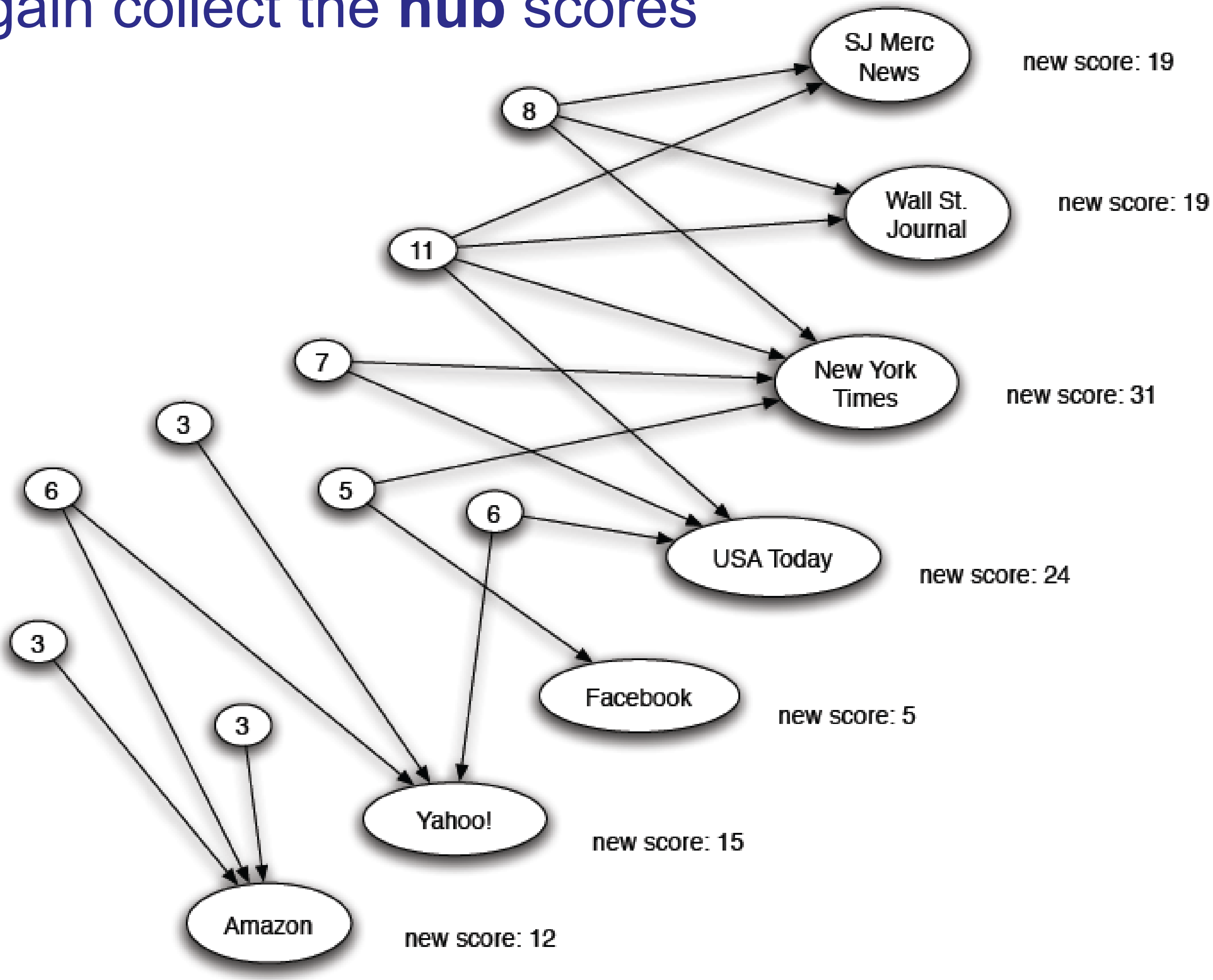
Sum of authority scores of nodes that the node points to.



- **Hubs** collect authority scores

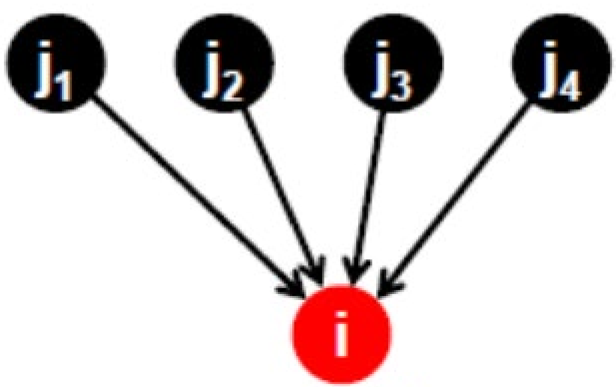
Mutual (recursive) Reinforcement

- **Authorities** again collect the **hub** scores

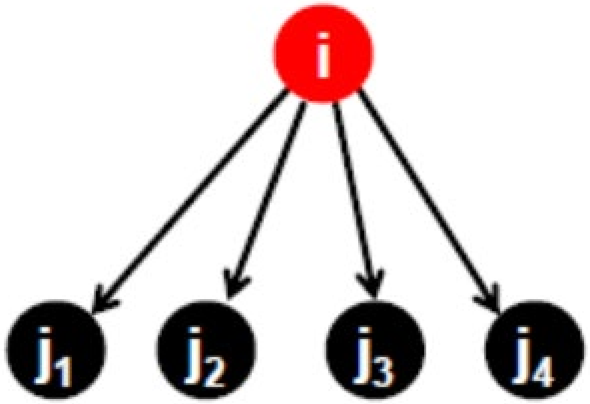


Computing Hubs and Authorities

- **Each page *i* has 2 scores**
  - Authority score: a_i
  - Hub score: h_i



$$a_i = \sum_{j \to i} h_j$$



$$h_i = \sum_{i \to j} a_j$$

Computing Hubs and Authorities

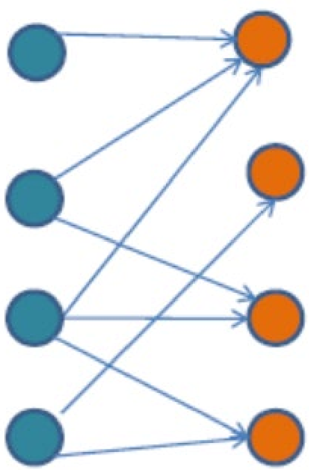
- **HITS is iterative**
  - Keep iterating until convergence
  - Mutual reinforcing relationship

$$
\begin{cases}
\forall \boldsymbol{i}\text{: Authority: } a_i^{(t+1)} = \sum_{j \to i} h_j^{(t)} \\
\forall \boldsymbol{i}\text{: Hub: } h_i^{(t+1)} = \sum_{i \to j} a_j^{(t)}
\end{cases}
$$

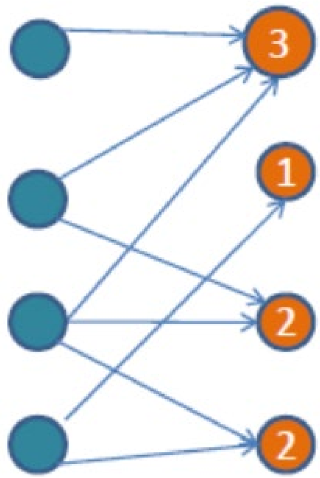
Example

What's the problem?

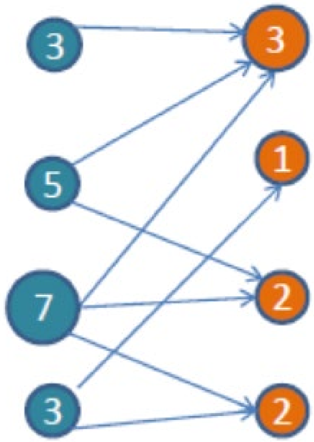
Initial graph

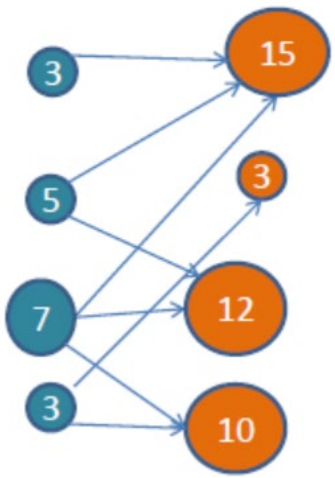


Iteration 1: Compute a

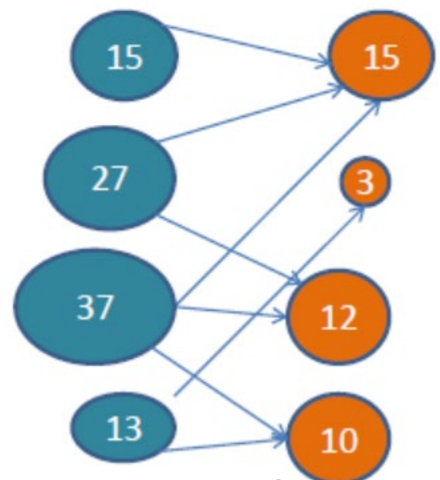


Iteration 1: Compute h





Iteration 2: Compute a



Iteration 2: Compute h

Need to Normalize Scores!

Initialize: $a_j^{(0)} = 1/\sqrt{\mathrm{N}}$, $\mathrm{h}_\mathrm{j}^{(0)} = 1/\sqrt{\mathrm{N}}$

Iterate till convergence

$\forall \boldsymbol{i}$ **compute a and h**

$\forall \boldsymbol{i}$: Normalize:

$\sum_i \left(a_i^{(t+1)}\right)^2 = 1, \sum_j \left(h_j^{(t+1)}\right)^2 = 1$

Does HITS converge?

- HITS converges to a single stable point
- Notations:

  Vector $\boldsymbol{a} = (a_1 \dots, a_n)$, $\boldsymbol{h} = (h_1 \dots, h_n)$

  Adjacency matrix $\boldsymbol{A}$ (NxN): $\boldsymbol{A}_{ij} = 1$ if $i \rightarrow j$, 0 otherwise
- $\boldsymbol{h}_i = \sum_{i \rightarrow j} \boldsymbol{a}_j$ can be rewritten as $\boldsymbol{h}_i = \sum_j \boldsymbol{A}_{ij} \cdot \boldsymbol{a}_j$
- We have $\boldsymbol{h} = \boldsymbol{A} \cdot \boldsymbol{a}$
- $\boldsymbol{a}_i = \sum_{j \rightarrow i} \boldsymbol{h}_j$ can be rewritten as

  $$\boldsymbol{a}_i = \sum_j \boldsymbol{A}_{ji} \cdot \boldsymbol{h}_j = \boldsymbol{A}^T \cdot \boldsymbol{h}$$
- We have $\boldsymbol{a} = \boldsymbol{A}^T \cdot \boldsymbol{h}$

Hubs and Authorities

- **HITS algorithm in vector notation**

  Set: $\boldsymbol{a}_i = \boldsymbol{h}_i = \frac{\mathbf{1}}{\sqrt{\boldsymbol{n}}}$

- **Repeat until convergence**

  $$\boldsymbol{h} = \boldsymbol{A} \cdot \boldsymbol{a}$$

  $$\boldsymbol{a} = \boldsymbol{A}^{\boldsymbol{T}} \cdot \boldsymbol{h}$$

  Normalize $\boldsymbol{a}$ and $\boldsymbol{h}$

Convergence criterion:

$$\sum_i \left(h_i^{(t)} - h_i^{(t-1)}\right)^2 < \varepsilon$$

$$\sum_i \left(a_i^{(t)} - a_i^{(t-1)}\right)^2 < \varepsilon$$

- **Now** $\boldsymbol{a} = \underbrace{\boldsymbol{A}^{\boldsymbol{T}} \cdot (\underbrace{\boldsymbol{A} \cdot \boldsymbol{a}}_{\text{new } h})}_{\text{new } a}$

a is updated (in 2 steps):

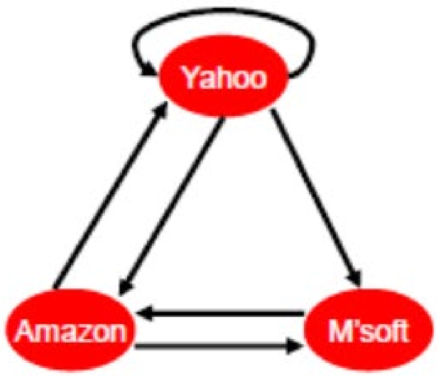
$$a = A^T(A\,a) = (A^T A)\,a$$

h is updated (in 2 steps):

$$h = A\,(A^T h) = (A\,A^T)\,h$$

Example of HITS

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



h(yahoo) = .58
h(amazon) = .58
h(m'soft) = .58

a(yahoo) = .58
a(amazon) = .58
a(m'soft) = .58

PageRank vs HITS

- **PageRank and HITS are two solutions to same problem**
  - Make use of link structure to rank pages
- **Which one is more suitable for large dataset?**
  - PageRank computed for all web pages and stored prior to query
  - HITS operates on a (query dependent) small subgraph from web graph
- **What is the value of an in-link from *u* to *v*?**
  - In the PageRank model, the value of the link depends on the **links into *u***
  - In the HITS model, it depends on the value of the other **links out of *u*** (vulnerable to spam)

Summary

- Link analysis in social network graphs to find communities
- Girvan-Newman algorithm use edge betweenness measure to separate nodes into communities

- Content of web pages and hyperlinks are important in web search
- Page Rank algorithm determine importance of web pages
- Trust Rank algorithm to overcome link spams