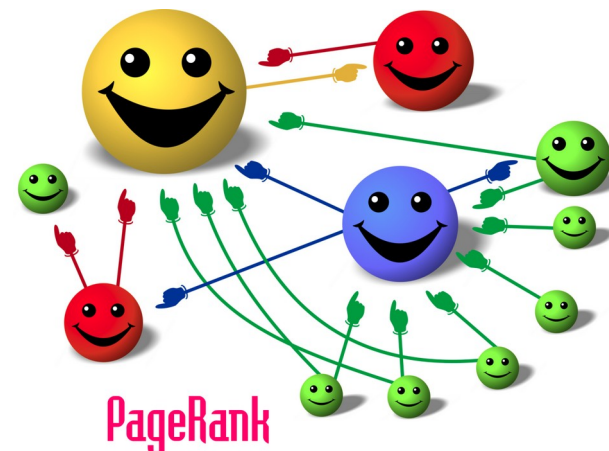
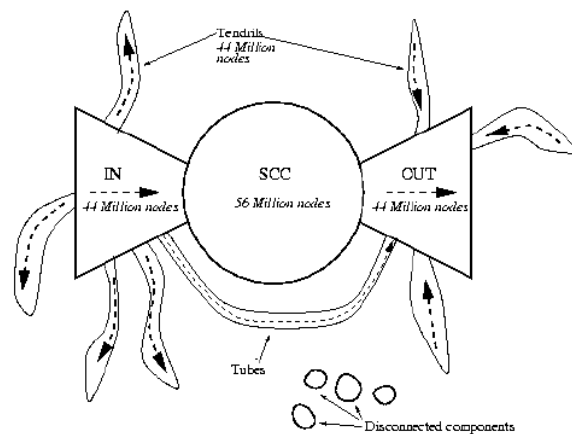


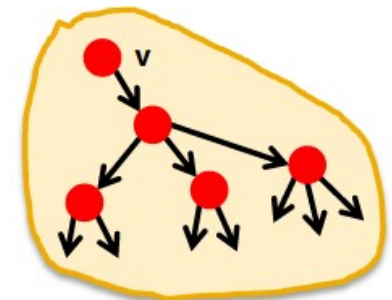
# Link Analysis: PageRank



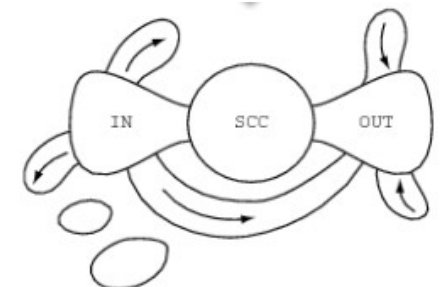
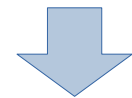
# Web as a Graph

# Structure of the Web

- On this lecture we will talk about how does the **Web graph** look like:



*Out(v)*



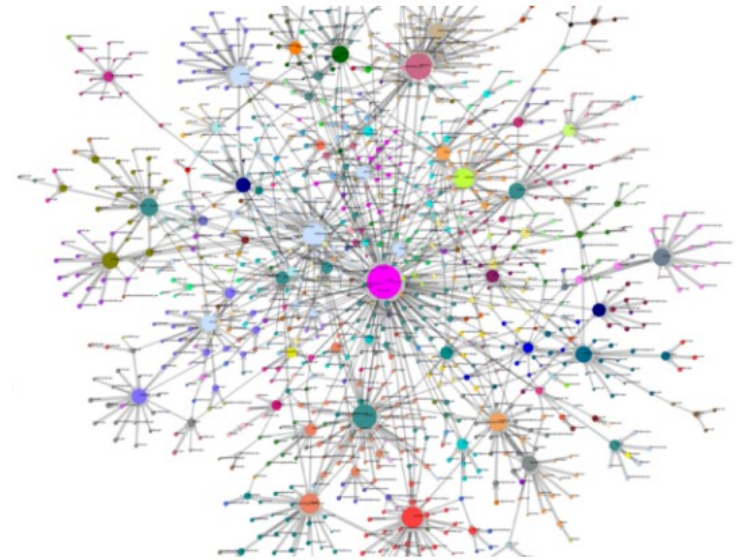
- 1) We will take a real system: **the Web**
- 2) We will represent it as a **directed graph**
- 3) We will use the language of **graph theory**
  - Strongly Connected Components
- 4) We will design a **computational experiment**:
  - Find In-and Out-components of a given node  $v$
- 5) We will learn something about the **structure of the Web: BOWTIE!**

# The Web as a Graph

**Q:** what does the Web “look like” at a global level?

- Web as a **graph**:

- Nodes = web pages
- Edges = hyperlinks



- Side issue: **what is a node?**

- Dynamic pages created on the fly
- “dark matter” – inaccessible database generated pages

# The Web as a Graph: Example

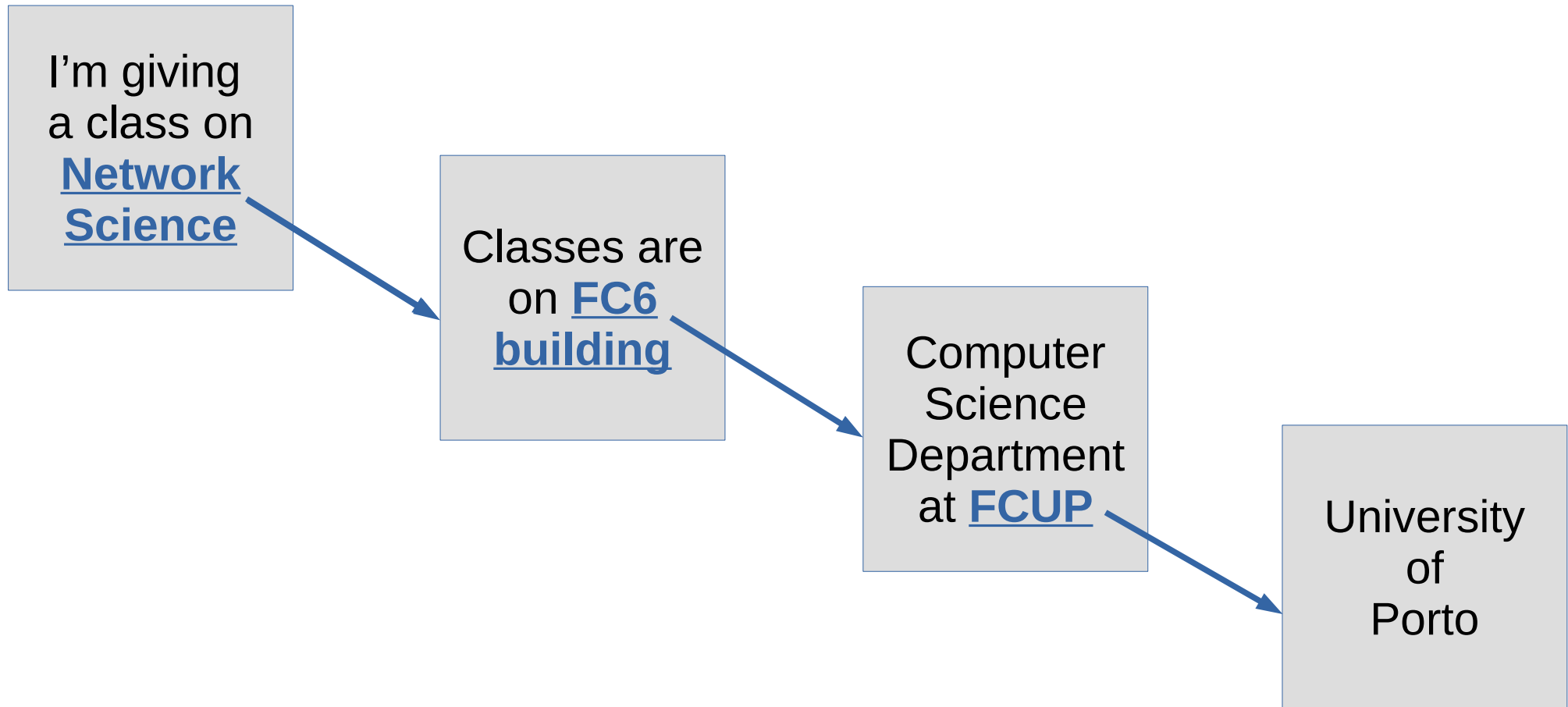
I'm giving  
a class on  
Network  
Science

Classes are  
on FC6  
building

Computer  
Science  
Department  
at FCUP

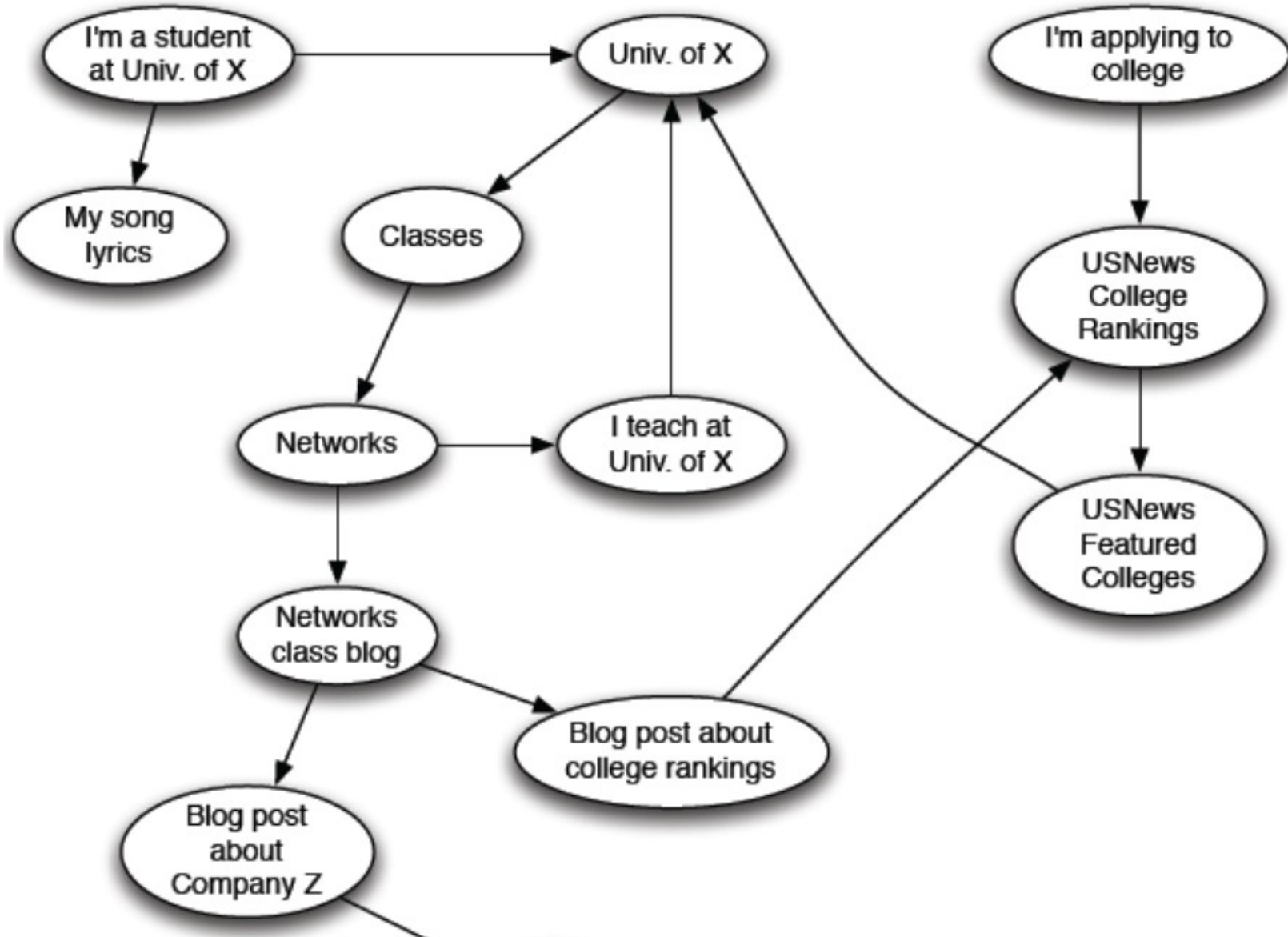
University  
of  
Porto

# The Web as a Graph: Example

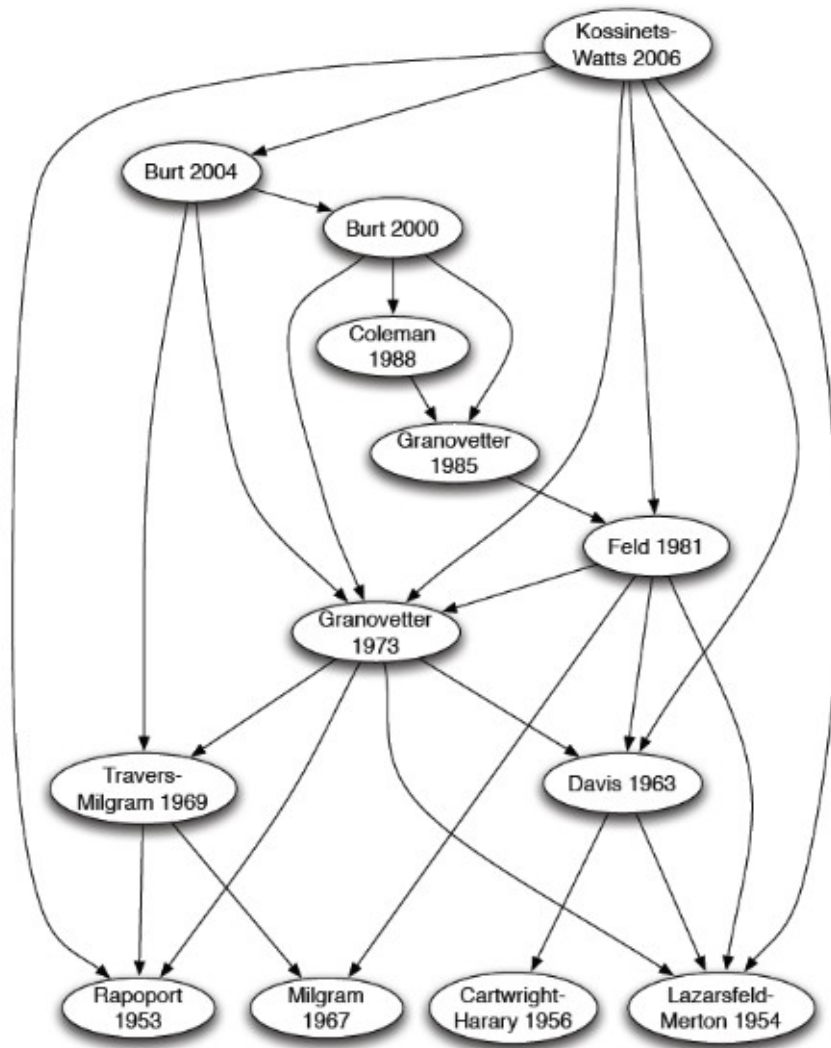


- In early days of the Web links were **navigational**
- Today many links are **transactional** (used not to navigate from page to page, but to post, comment, like, buy, ...)

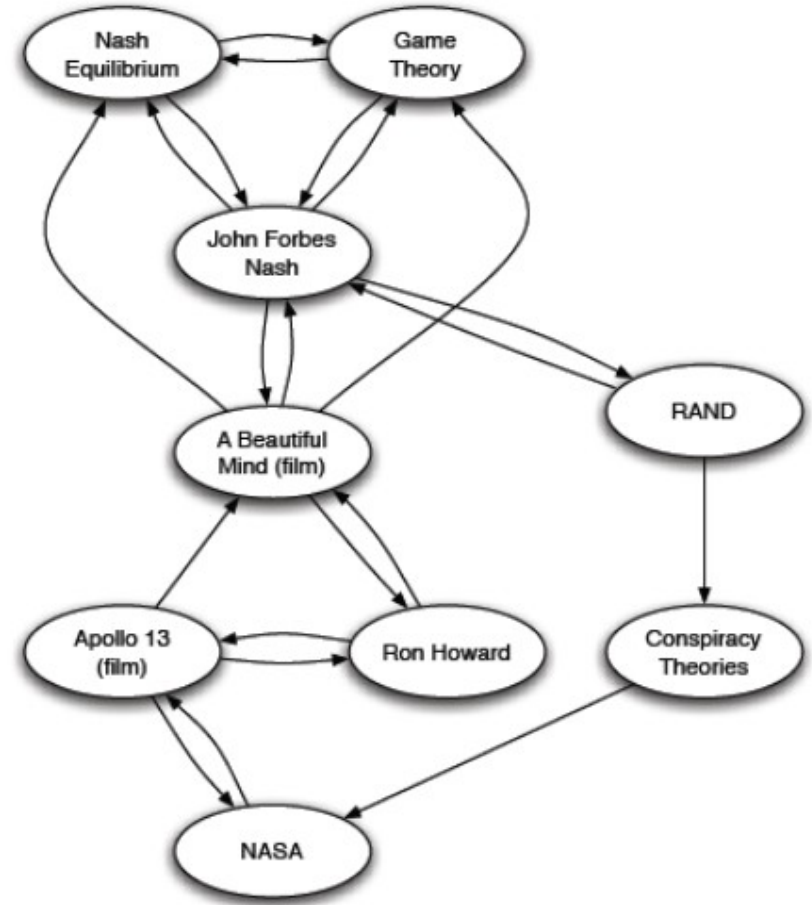
# The Web as a Directed Graph



# Other Information Networks



Citations



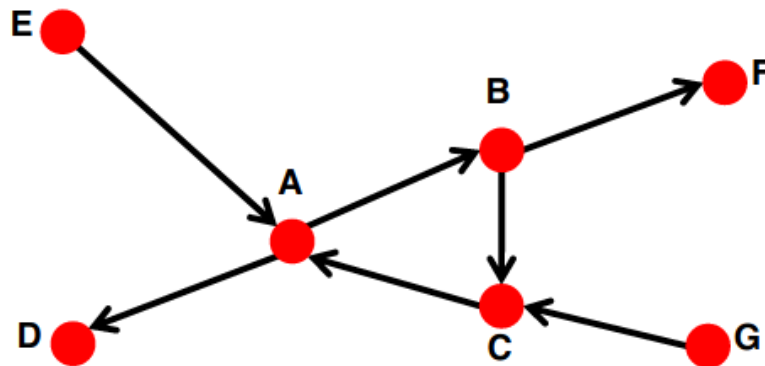
Wikipedia

# What does the Web look like?

- How is the Web linked?
- What is the “map” of the Web?

Web as a **directed graph** [Broder et al. 2000]:

- Given node **v**, what nodes can **v** reach?
- What other nodes can reach **v**?



$$In(v) = \{w \mid w \text{ can reach } v\}$$

$$Out(v) = \{w \mid v \text{ can reach } w\}$$

For example:

$$In(A) = \{A, B, C, E, G\}$$

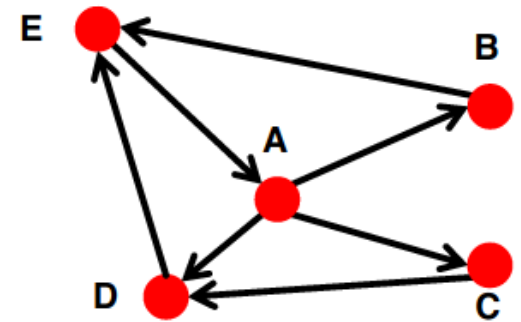
$$Out(A) = \{A, B, C, D, F\}$$

# Reasoning About Directed Graphs

- Two types of directed graphs:

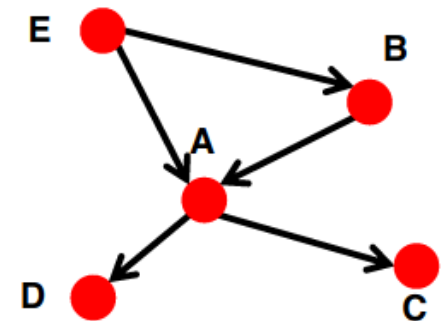
- **Strongly connected:**

- Any node can reach any node via a directed path  
 $In(A) = Out(A) = \{A, B, C, D, E\}$



- **Directed Acyclic Graph (DAG):**

- Has no cycles: if  $u$  can reach  $v$ , then  $v$  cannot reach  $u$

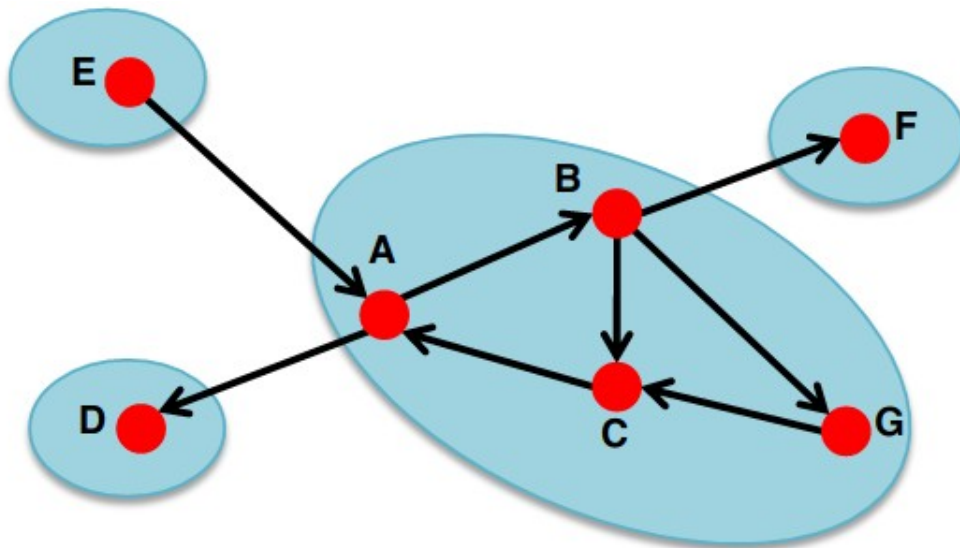


- **Any directed graph (the Web) can be expressed in terms of these two types!**

- Is the Web a big strongly connected graph or a DAG?

# Strongly Connected Component

- A **Strongly Connected Component** (SCC) is a set of nodes  $S$  so that:
  - Every pair of nodes in  $S$  can reach each other
  - There is no larger set containing  $S$  with this property



Strongly connected components of the graph: {A,B,C,G}, {D}, {E}, {F}

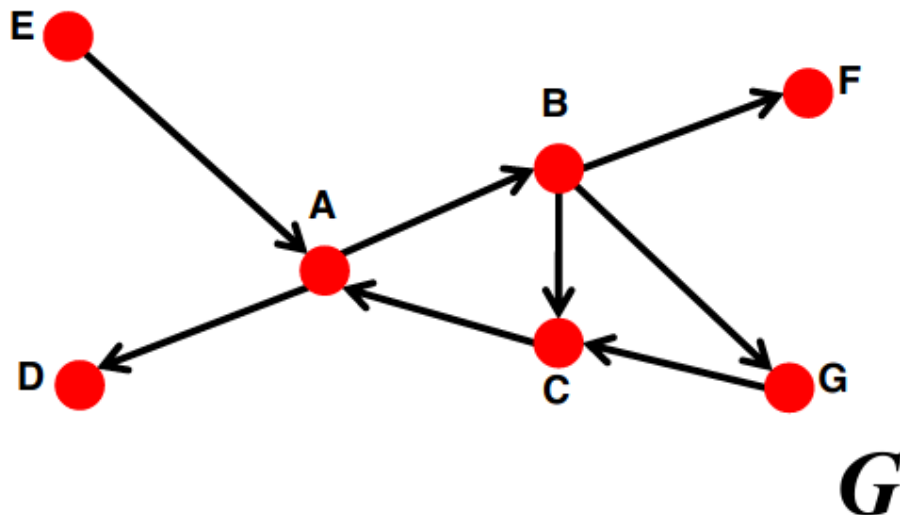
# Strongly Connected Component

- Fact: **Every directed graph is a DAG on its SCCs**

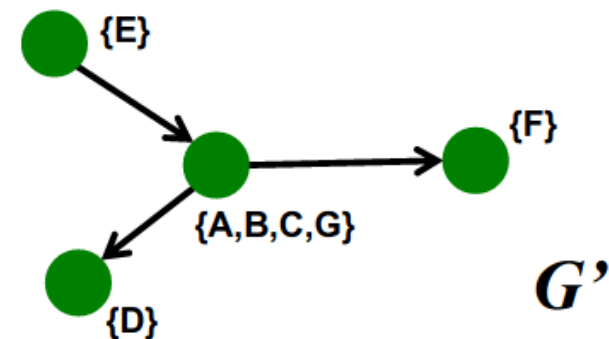
1) SCCs partition the nodes of  $G$

- That is, each node is in exactly one SCC

2) If we build a graph  $G'$  whose nodes are SCCs, and with an edge between nodes of  $G'$  if there is an edge between corresponding SCCs in  $G$ , then  $G'$  is a DAG



(1) Strongly connected components of graph  $G$ :  $\{A, B, C, G\}$ ,  $\{D\}$ ,  $\{E\}$ ,  $\{F\}$   
(2)  $G'$  is a DAG:



# Structure of the Web

- **Broder et al.:** Altavista web crawl (Oct '99)
  - Web crawl is based on a large set of starting points accumulated over time from various sources, including voluntary submissions.
  - 203 million URLs and 1.5 billion links

**Goal: Take a large snapshot of the Web and try to understand how its SCCs “fit together” as a DAG**

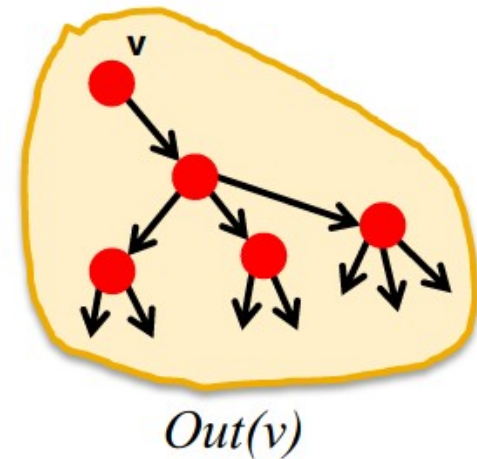


Tomkins,  
Broder, and  
Kumar

# Graph Structure of the Web

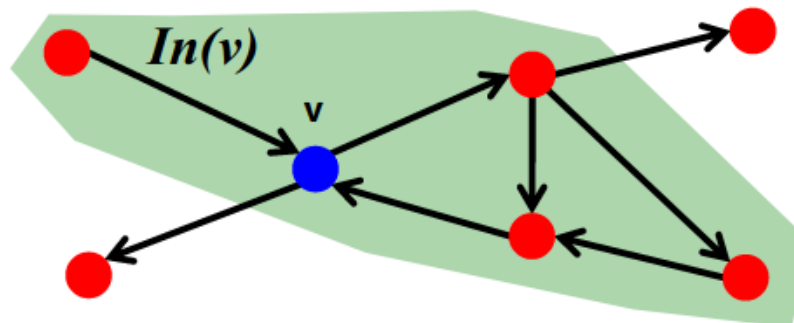
- **Computational issue:**

- Want to find a SCC containing node  $v$ ?



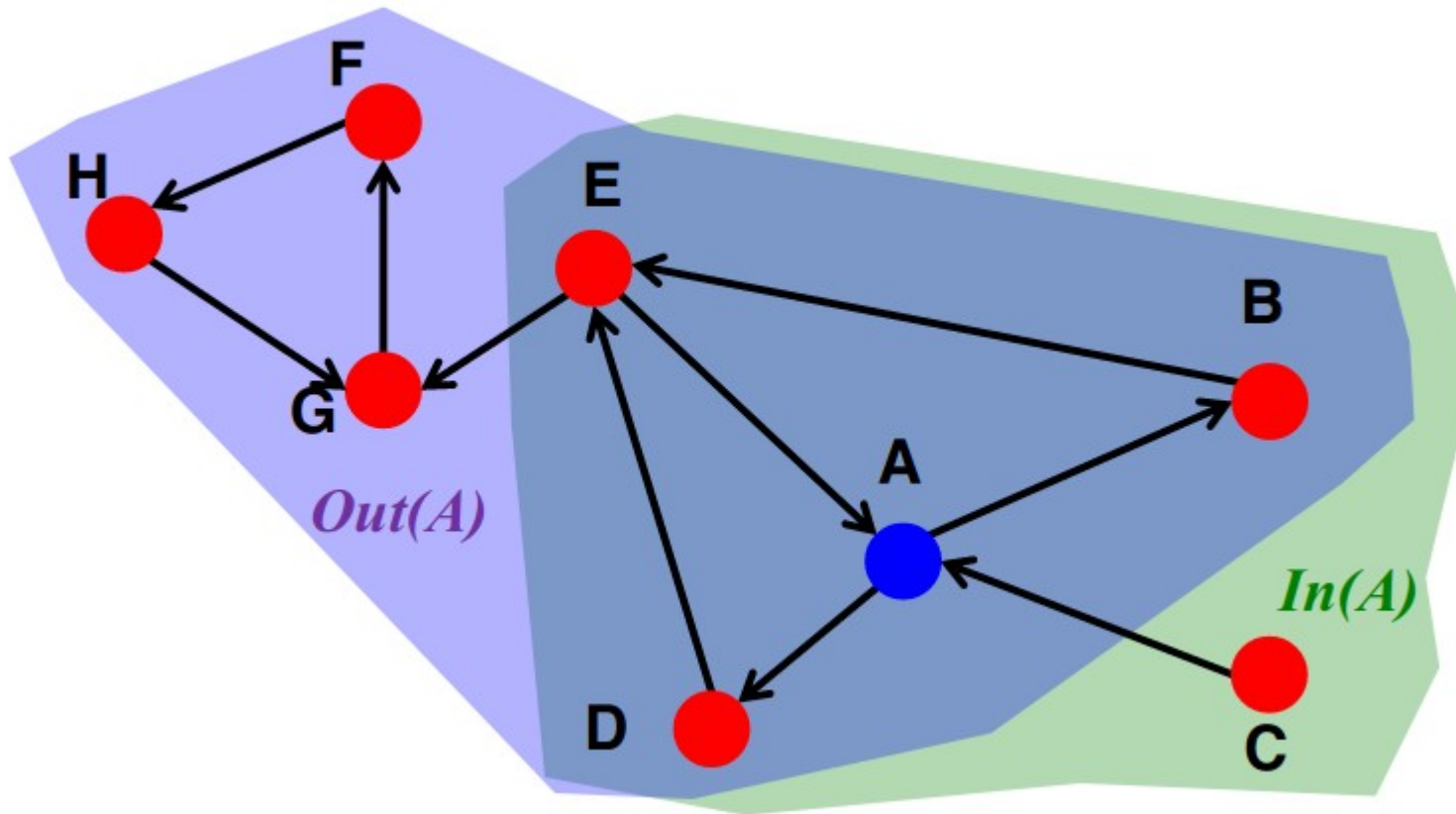
- **Observation:**

- **$Out(v)$**  ... nodes that can be reached from  $v$  (w/BFS)
- SCC containing  $v$  is:  
 $Out(v) \cap In(v) = Out(v, G) \cap Out(v, G')$ ,  
where  $G'$  is  $G$  with all edge directions flipped



# $Out(v) \cap In(v) = SCC$

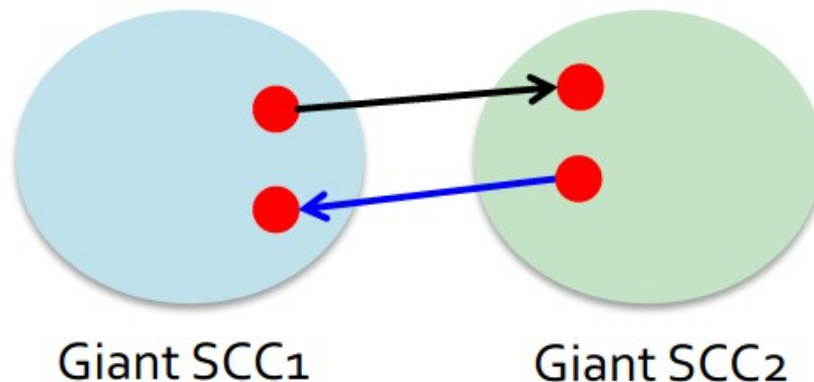
- Example:



- $Out(A) = \{A, B, D, E, F, G, H\}$
- $In(A) = \{A, B, C, D, E\}$
- So,  $SCC(A) = Out(A) \cap In(A) = \{A, B, D, E\}$

# Graph Structure of the Web

- **There is a single giant SCC**
  - That is, there won't be two SCCs
- **Why only 1 big SCC? Heuristic argument:**
  - Assume two equally big SCCs.
  - It just takes 1 page from one SCC to link to the other SCC.
  - If the two SCCs have millions of pages the likelihood of this not happening is very very small.

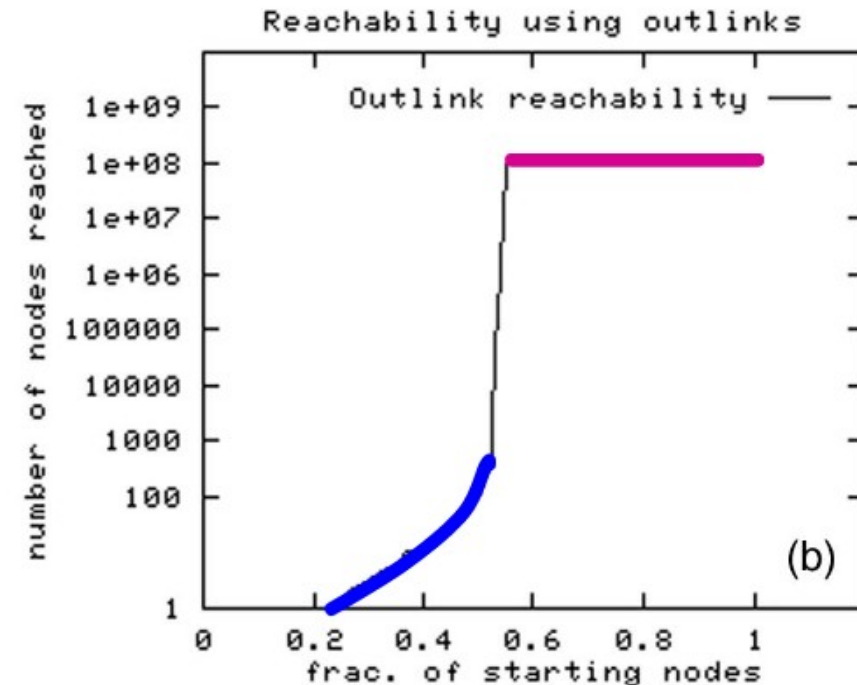


# Structure of the Web

- Directed version of the Web graph:
  - Altavista crawl from October 1999
    - 203 million URLs, 1.5 billion links

## Computation:

- Compute  $In(v)$  and  $Out(v)$  by starting at random nodes.
- **Observation:** The BFS either visits **many nodes** or **very few**

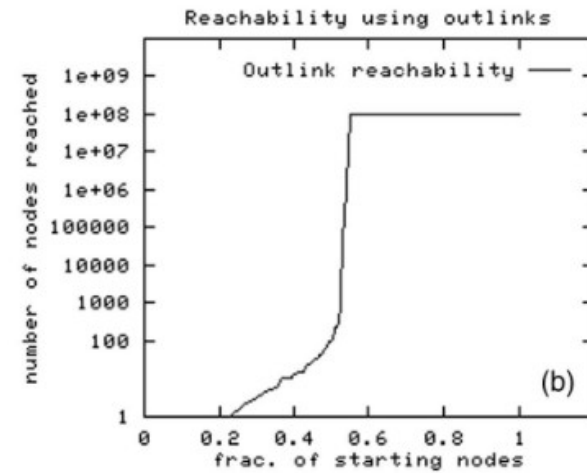


x-axis: rank

y-axis: number of reached nodes

# Structure of the Web

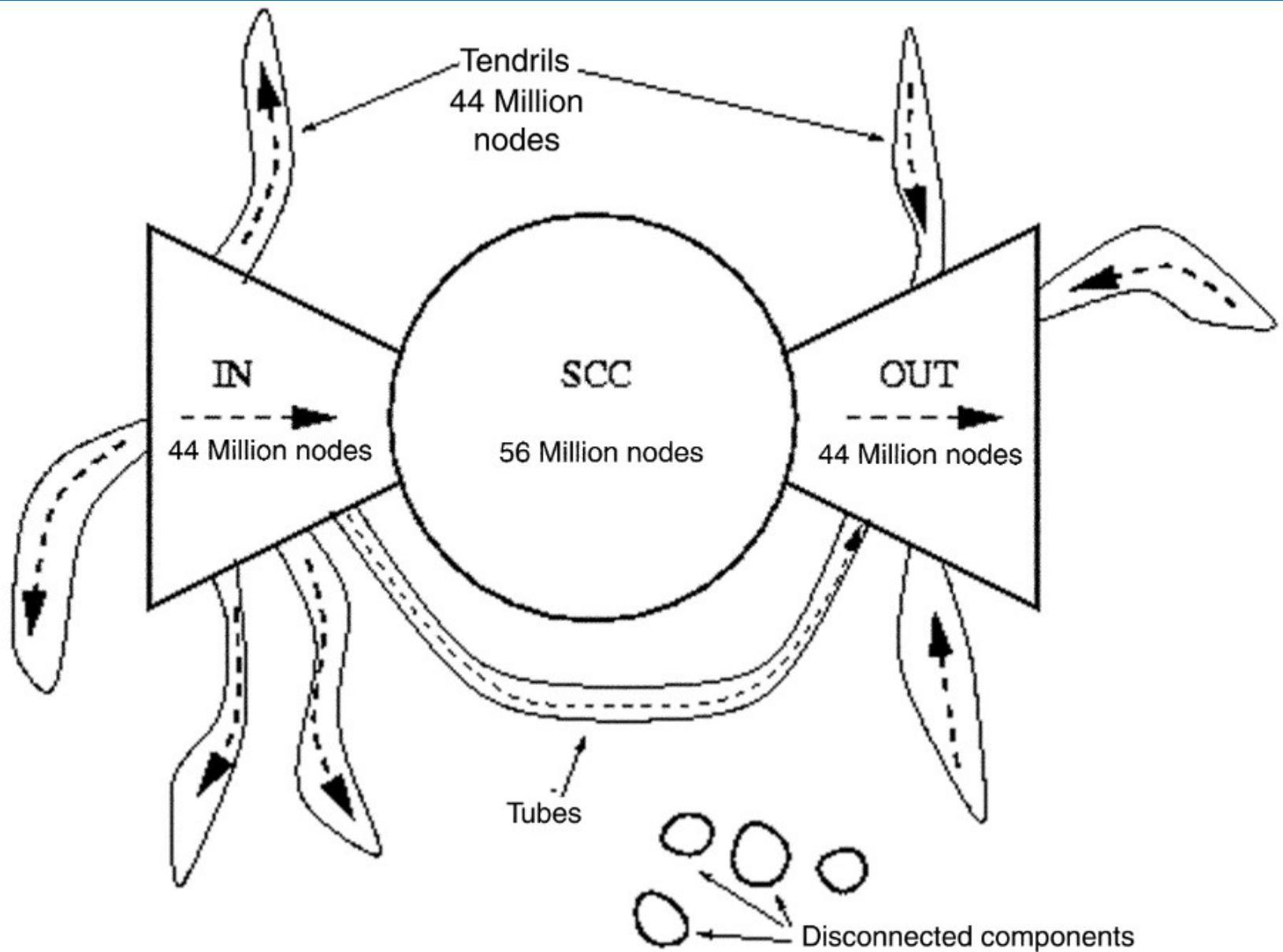
- **Result:** Based on IN and OUT of a random node  $v$ :
  - **$Out(v)$**   $\approx$  100 million (**50%** nodes)
  - **$In(v)$**   $\approx$  100 million (**50%** nodes)
  - **Largest SCC:** 56 million (**28%** nodes)



x-axis: rank  
y-axis: number of reached nodes

- **What does this tell us about the conceptual picture of the Web Graph?**

# Bowtie Structure of the Web



203 million pages, 1.5 billion links [Broder et al. 2000]

# **How to Organize the Web?**

## **Link Analysis**

# How to Organize the Web?

- How to organize the Web?

- First try: Human curated

## Web directories

- Yahoo, Sapo



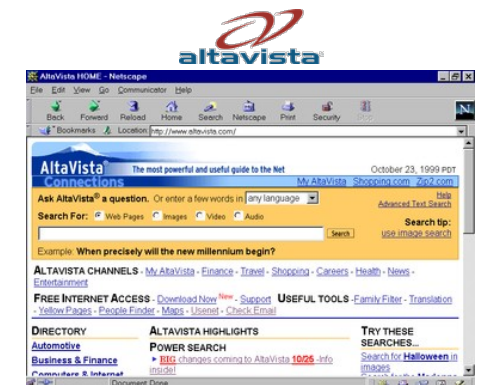
- Second try: **Web Search**

- **Information Retrieval:** attempts to find relevant docs in a small and trusted set

- Newspaper articles, Patents, etc.

- **But:** Web is huge, full of untrusted documents, random things, spam, etc.

- **So we need a good way to rank webpages!**



# Web Search: Challenges

2 challenges of web search

## 1) **Web contains many sources of information**

Who to “trust”?

- **Insight:** Trustworthy pages may point to each other!

## 2) **What is the “best” answer to query “newspaper”**

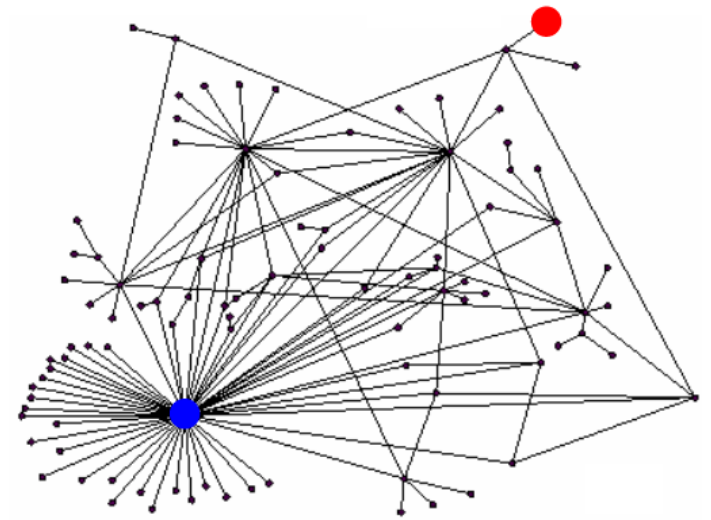
- No single right answer

- **Insight:** Pages that actually know about newspapers might all be pointing to many newspapers

# Ranking Nodes on the Graph

- Web pages are not equally “important”
  - [www.joe-nobody.com](http://www.joe-nobody.com) vs [www.up.pt](http://www.up.pt)

- We already know: There is a large diversity in the web graph node connectivity



- **So, let's rank the pages using the web graph link structure!**

# Link Analysis Algorithms

- We will cover the following **Link Analysis** approaches to computing the importance of nodes in a graph:
  - Hubs and Authorities (**HITS**)
  - **PageRank**
  - Topic-Specific (**Personalized**) **PageRank**

**Sidenote:** Various notions of **node centrality**: Node  $u$

- ▣ **Degree centrality** = degree of  $u$
- ▣ **Betweenness centrality** = #shortest paths passing through  $u$
- ▣ **Closeness centrality** = avg. length of shortest paths from  $u$  to all other nodes of the network
- ▣ **Eigenvector centrality** = like PageRank

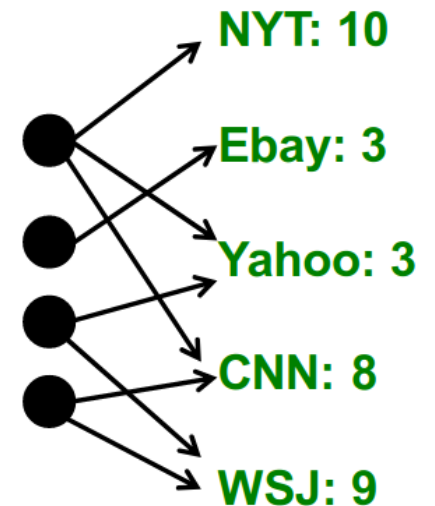
# **Hubs and Authorities (HITS)**

# Link Analysis

- Goal(back to the newspaper example):
  - Don't just find newspapers. Find “experts” – pages that link in a coordinated way to good newspapers
- Idea: **Links as votes**
  - Page is more important if it has more links
    - In-coming links? Out-going links?
- Hubs and Authorities

Each page has 2 scores:

  - Quality as an expert (**hub**):
    - Total sum of votes of pages pointed to
  - Quality as an content (**authority**):
    - Total sum of votes of experts
  - **Principle of repeated improvement**



# Hubs and Authorities

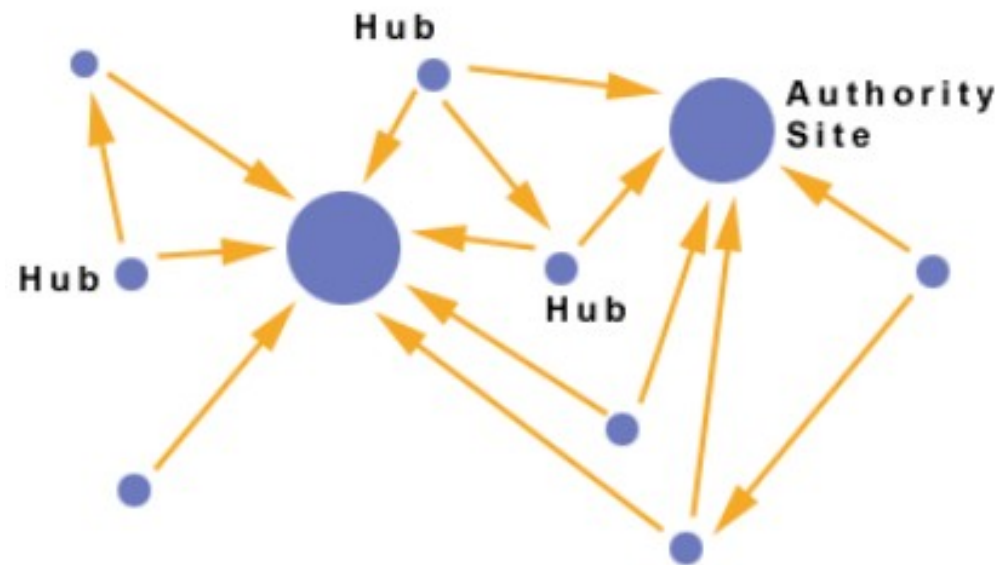
Interesting pages fall into two classes:

**1) Authorities** are pages containing useful information

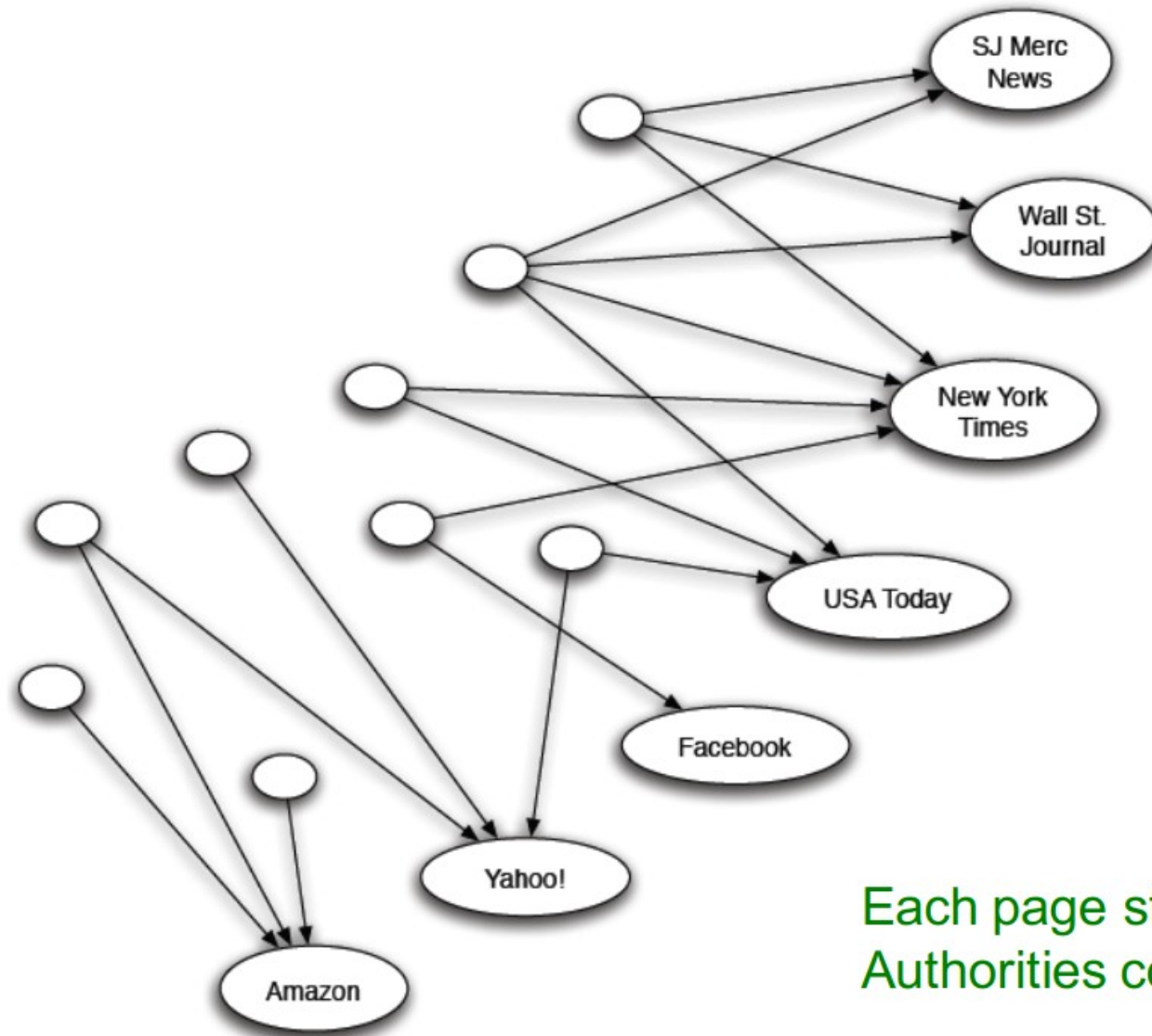
- Newspaper home pages
- Course home pages
- Home pages of auto manufacturers

**2) Hubs** are pages that link to authorities

- List of newspapers
- Course bulletin
- List of auto manufacturers



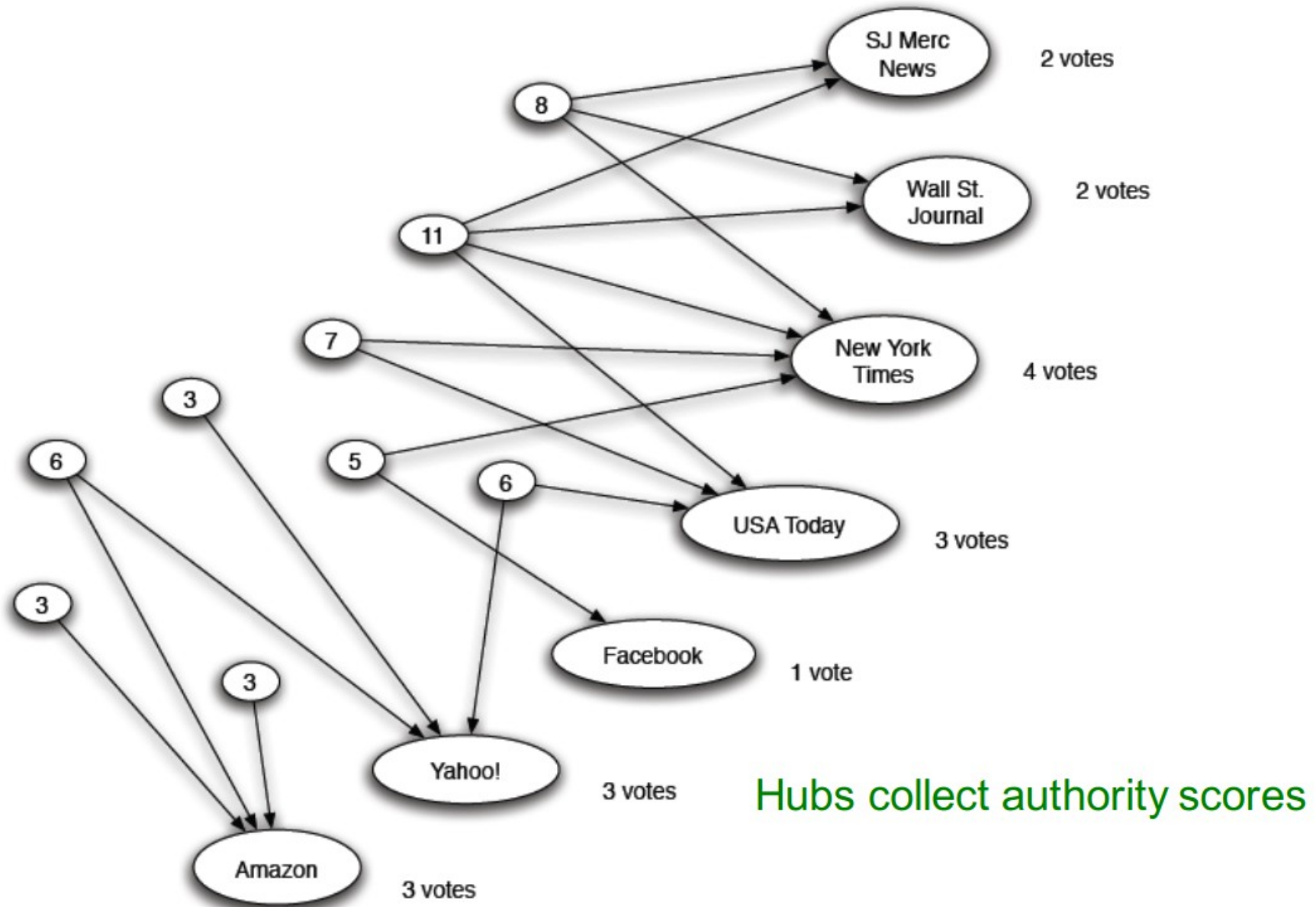
# Counting in-links: Authority



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

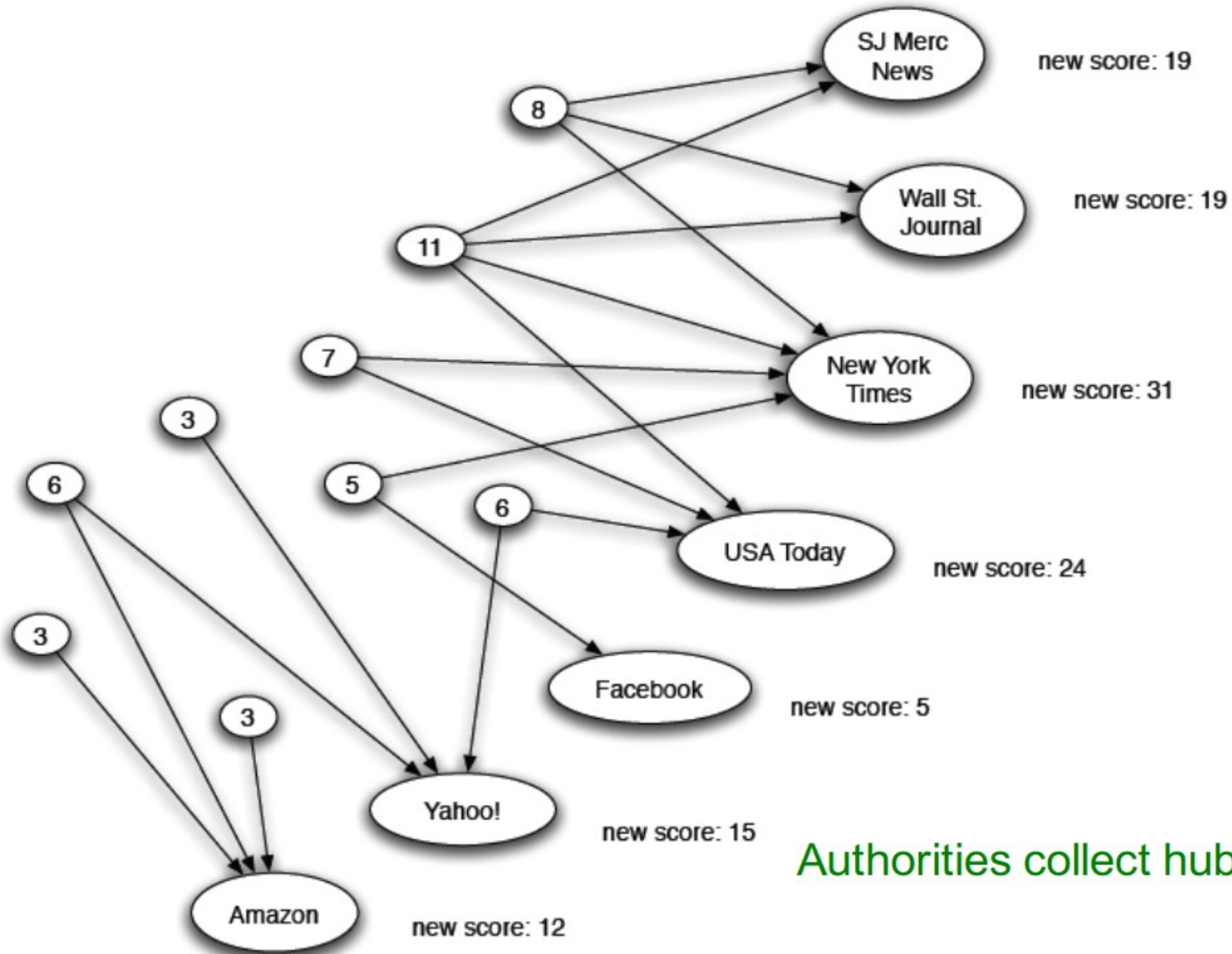
(Note this is idealized example. In reality graph is not bipartite and each page has both a hub and the authority score)

# Expert Quality: Hub



(Note this is idealized example. In reality graph is not bipartite and each page has both a hub and authority score)

# Reweighting



Authorities collect hub scores

(Note this is idealized example. In reality graph is not bipartite and each page has both a hub and authority score)

# Mutually Recursive Definition

- A good **hub** links to many good authorities
- A good **authority** is linked from many good hubs
  - Note a self-reinforcing recursive definition
- Model using **two scores for each node**:
  - **Hub** score and **Authority** score
  - Represented as vectors  $h$  and  $a$ , where the  $i$ -th element is the hub/authority score of the  $i$ -th node

# Hubs and Authorities

- **Each page  $i$  has 2 scores:**

- Authority score:  $a_i$

- Hub score:  $h_i$

**Convergence criteria:**

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

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

## HITS algorithm:

- Initialize:  $a_j^{(0)} = 1/\sqrt{n}$ ,  $h_j^{(0)} = 1/\sqrt{n}$

- Then keep iterating until **convergence**:

- $\forall i$ : Authority:  $a_i^{(t+1)} = \sum_{j \rightarrow i} h_j^{(t)}$

- $\forall i$ : Hub:  $h_i^{(t+1)} = \sum_{i \rightarrow j} a_j^{(t)}$

- $\forall i$ : Normalize:

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

## □ Hits in the vector notation:

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

□ Adjacency matrix  $A$  ( $n \times n$ ):  $A_{ij} = 1$  if  $i \rightarrow j$

□ **Can rewrite**  $h_i = \sum_{i \rightarrow j} a_j$  **as**  $h_i = \sum_j A_{ij} \cdot a_j$

□ **So:**  $\mathbf{h} = A \cdot \mathbf{a}$  And similarly:  $\mathbf{a} = A^T \cdot \mathbf{h}$

## □ Repeat until convergence:

□  $h^{(t+1)} = A \cdot a^{(t)}$

□  $a^{(t+1)} = A^T \cdot h^{(t)}$

□ Normalize  $a^{(t+1)}$  and  $h^{(t+1)}$

□ What is  $a = A^T \cdot h$ ?

□ Then:  $a = A^T \cdot \underbrace{(A \cdot a)}_{\text{new } h}$   
 $\underbrace{\hspace{10em}}_{\text{new } a}$

□  $a$  is updated (in 2 steps):

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

□  $h$  is updated (in 2 steps)

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

□ Thus, in  $2k$  steps:

$$a = (A^T \cdot A)^k \cdot a$$

$$h = (A \cdot A^T)^k \cdot h$$

Repeated matrix powering

## □ Definition: Eigenvectors & Eigenvalues

□ Let  $R \cdot x = \lambda \cdot x$

for some scalar  $\lambda$ , vector  $x$ , matrix  $R$

□ Then  $x$  is an **eigenvector**, and  $\lambda$  is its **eigenvalue**

## □ The steady state (HITS has converged):

□  $A^T \cdot A \cdot a = c' \cdot a$

□  $A \cdot A^T \cdot h = c'' \cdot h$

- So, **authority**  $a$  is eigenvector of  $A^T A$   
(associated with the largest eigenvalue)  
Similarly: **hub**  $h$  is eigenvector of  $AA^T$

Note constants  $c', c''$   
don't matter as we  
normalize them out  
every step of HITS

# PageRank

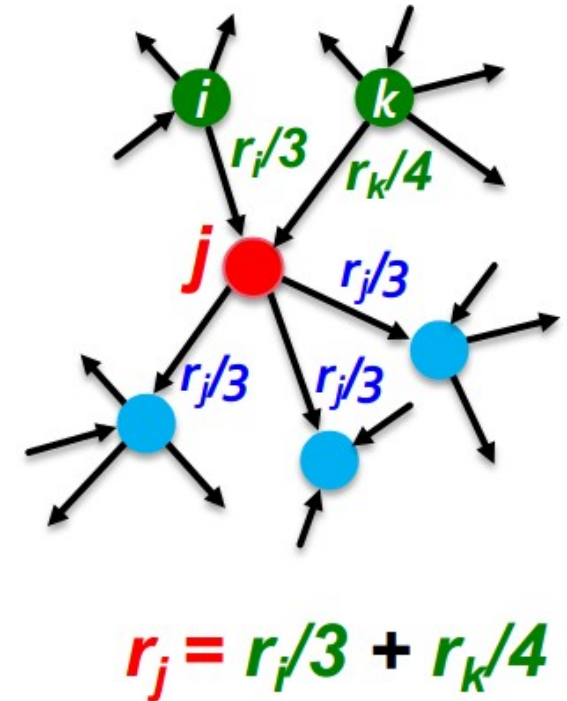
(a.k.a., the Google Algorithm)

# Links as Votes

- **Still the same idea: Links as votes**
  - Page is more important if it has more links
    - In-coming links? Out-going links?
- **Think of in-links as votes:**
  - [www.up.pt](http://www.up.pt) has 42,000 in-links
  - [www.joe-nobody.com](http://www.joe-nobody.com) has 1 in-link
- **Are all in-links equal?**
  - Links from important pages count more
  - Recursive question!

# PageRank: the “Flow” Model

- A “vote” from an important page is worth more:
  - Each link’s vote is proportional to the **importance** of its source page
  - If page  $i$  with importance  $r_i$  has  $d_i$  out-links, each link gets  $r_i / d_i$  votes
  - Page  $j$ ’s own importance  $r_j$  is the sum of the votes on its in-links

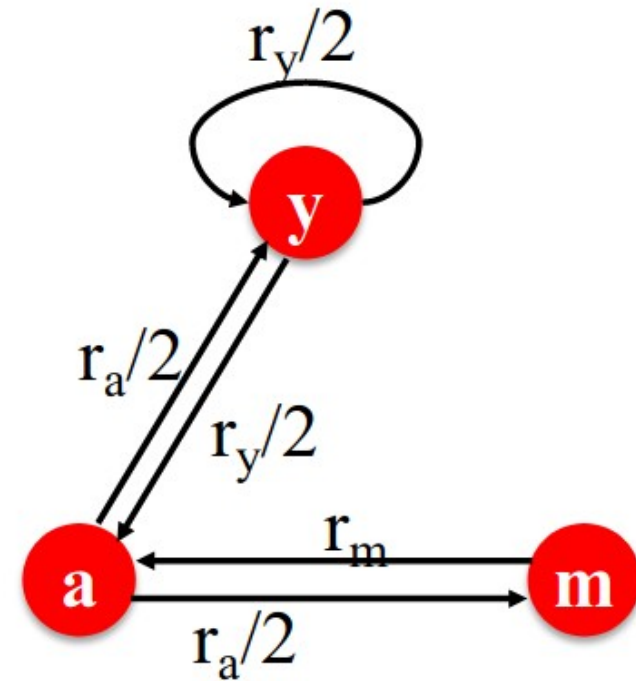


# PageRank: the “Flow” Model

- A page is important if it is pointed to by other important pages
- Define a “rank”  $r_j$  for node  $j$

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

$d_i$  ... out-degree of node  $i$



“Flow” equations:

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

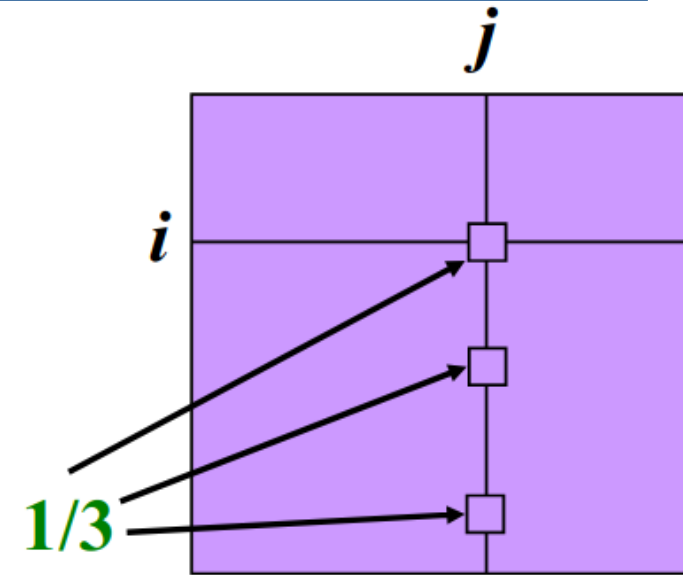
$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

# PageRank: Matrix Formulation

## ■ Stochastic adjacency matrix $M$

- Let page  $j$  have  $d_j$  out-links
- If  $j \rightarrow i$ , then  $M_{ij} = \frac{1}{d_j}$ 
  - $M$  is a **column stochastic matrix**
    - Columns sum to 1



## ■ Rank vector $r$ : An entry per page

- $r_i$  is the importance score of page  $i$
- $\sum_i r_i = 1$

## ■ The flow equations can be written

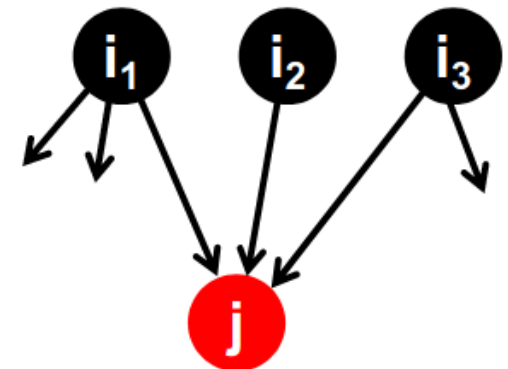
$$r = M \cdot r$$

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

# Random Walk Interpretation

## ■ Imagine a random web surfer:

- At any time  $t$ , surfer is on some page  $i$
- At time  $t + 1$ , the surfer follows an out-link from  $i$  uniformly at random
- Ends up on some page  $j$  linked from  $i$
- Process repeats indefinitely



$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_{\text{out}}(i)}$$

## ■ Let:

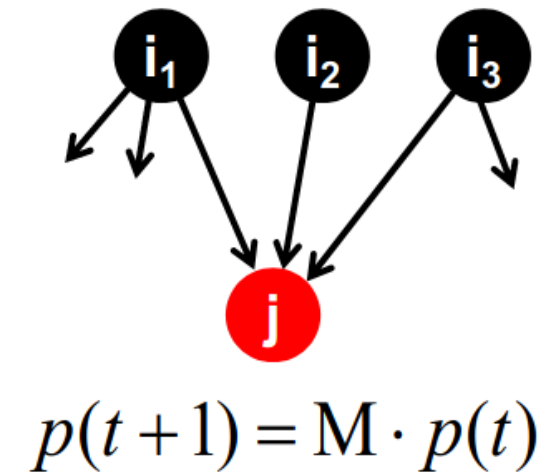
- $\mathbf{p}(t)$  ... vector whose  $i^{\text{th}}$  coordinate is the prob. that the surfer is at page  $i$  at time  $t$
- So,  $\mathbf{p}(t)$  is a probability distribution over pages

# The Stationary Distribution

- **Where is the surfer at time  $t+1$ ?**

- Follows a link uniformly at random

$$p(t+1) = M \cdot p(t)$$



- Suppose the random walk reaches a state

$$p(t+1) = M \cdot p(t) = p(t)$$

then  $p(t)$  is **stationary distribution** of a random walk

- **Our original rank vector  $r$**  satisfies  $r = M \cdot r$

- **So,  $r$  is a stationary distribution for the random walk**

# PageRank

## How to Solve?

# PageRank: How to Solve?

Given a web graph with  $n$  nodes, where the nodes are pages and edges are hyperlinks

- Assign each node an initial page rank
- Repeat until convergence ( $\sum_i |r_i^{(t+1)} - r_i^{(t)}| < \epsilon$ )
  - Calculate the page rank of each node

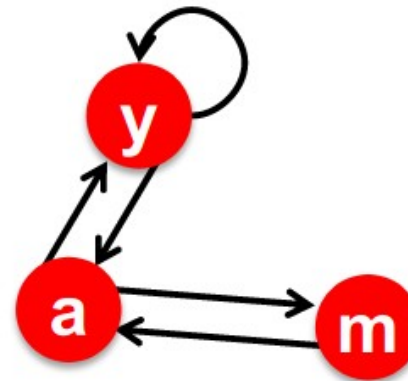
$$r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{d_i}$$

$d_i$  .... out-degree of node  $i$

# PageRank: How to Solve?

## ■ Power Iteration:

- Set  $r_j \leftarrow 1/N$
- **1:**  $r'_j \leftarrow \sum_{i \rightarrow j} \frac{r_i}{d_i}$
- **2:**  $r \leftarrow r'$
- If  $|r - r'| > \varepsilon$ : goto **1**



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	1
m	0	$\frac{1}{2}$	0

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

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

## ■ Example:

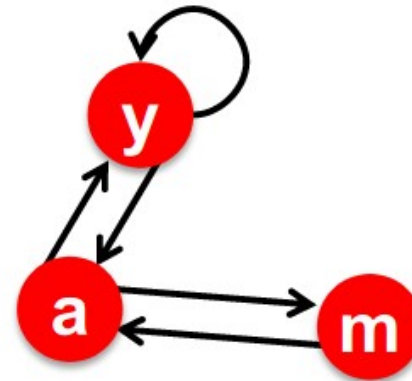
$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix}$$

Iteration 0, 1, 2, ...

# PageRank: How to Solve?

## ■ Power Iteration:

- Set  $r_j \leftarrow 1/N$
- **1:**  $r'_j \leftarrow \sum_{i \rightarrow j} \frac{r_i}{d_i}$
- **2:**  $r \leftarrow r'$
- If  $|r - r'| > \varepsilon$ : goto **1**



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	1
m	0	$\frac{1}{2}$	0

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

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

## ■ Example:

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{matrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \dots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 3/15 \end{matrix}$$

Iteration 0, 1, 2, ...

# PageRank: 3 Questions

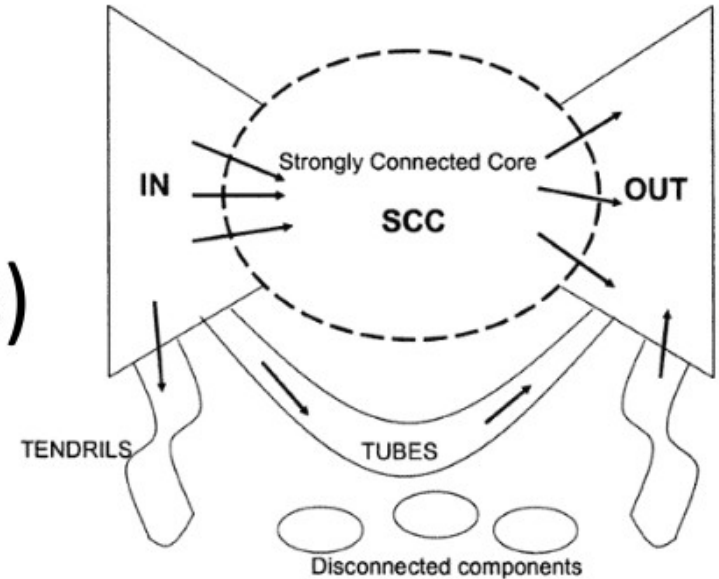
$$r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{d_i} \quad \text{or equivalently} \quad r = Mr$$

- Does this converge?
- Does it converge to what we want?
- Are the results reasonable?

# PageRank: Problems

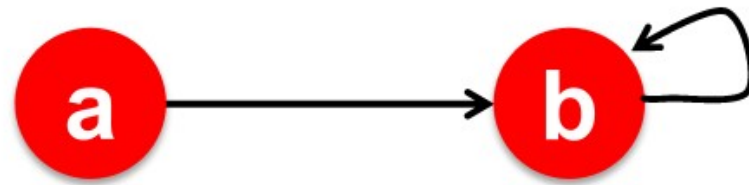
## Two problems:

- **(1)** Some pages are **dead ends** (have no out-links)
  - Such pages cause importance to “leak out”
- **(2) Spider traps**  
(all out-links are within the group)
  - Eventually spider traps absorb all importance



# Does it converge to what we want?

- The “Spider trap” problem:



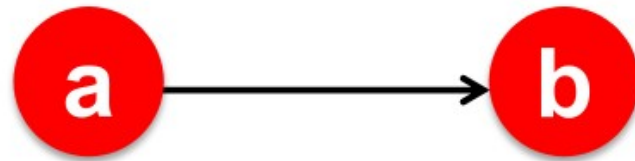
$$r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{d_i}$$

- Example:

	Iteration: 0,	1,	2,	3...
$r_a$	1	0	0	0
$r_b$	0	1	1	1

# Does it converge to what we want?

- The “Dead end” problem:



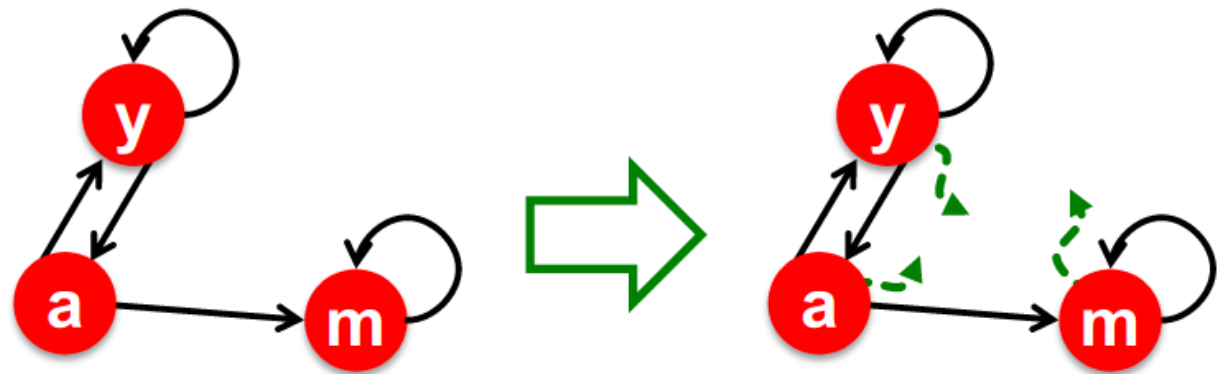
$$r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{d_i}$$

- Example:

	Iteration: 0,	1,	2,	3...
$r_a$	1	0	0	0
$r_b$	0	1	0	0

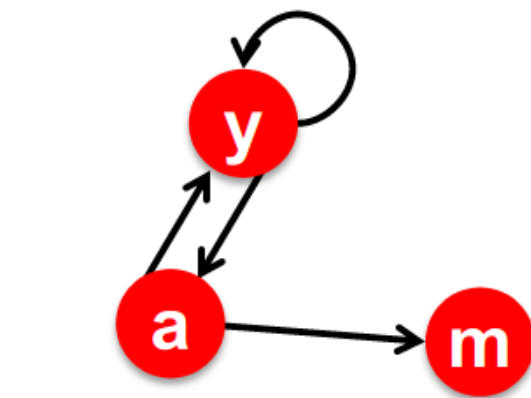
# Solution to Spider Traps

- **The Google solution for spider traps: At each time step, the random surfer has two options**
  - With prob.  $\beta$ , follow a link at random
  - With prob.  $1-\beta$ , jump to a random page
  - Common values for  $\beta$  are in the range 0.8 to 0.9
- **Surfer will teleport out of spider trap within a few time steps**

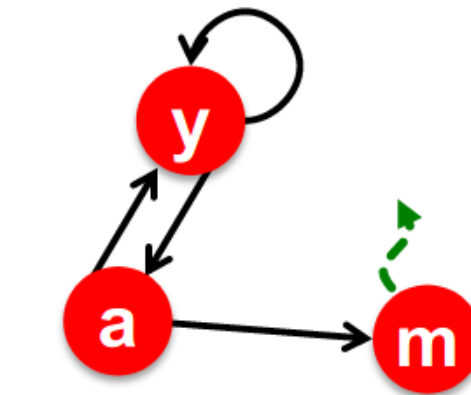


# Solution to Dead Ends

- **Teleports:** Follow random teleport links with probability **1.0** from dead-ends
  - Adjust matrix accordingly



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	0
m	0	$\frac{1}{2}$	0



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{3}$
a	$\frac{1}{2}$	0	$\frac{1}{3}$
m	0	$\frac{1}{2}$	$\frac{1}{3}$

# Final PageRank Equation

- **Google's solution:** At each step, random surfer has two options:
  - With probability  $\beta$ , follow a link at random
  - With probability  $1-\beta$ , jump to some random page
- **PageRank equation** [Brin-Page, '98]

$$r_j = \sum_{i \rightarrow j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{n}$$

$d_i$  ... out-degree  
of node  $i$

The above formulation assumes that  $M$  has no dead ends. We can either preprocess matrix  $M$  (**bad!**) or explicitly follow random teleport links with probability 1.0 from dead-ends. See P. Berkhin, *A Survey on PageRank Computing*, Internet Mathematics, 2005.

# The PageRank Algorithm

## ■ Input: Graph $G$ and parameter $\beta$

- Directed graph  $G$  with **spider traps** and **dead ends**
- Parameter  $\beta$

## ■ Output: PageRank vector $r$

- **Set:**  $r_j^{(0)} = \frac{1}{N}, \quad t = 1$

- **do:**

- $\forall j: r'_j^{(t)} = \sum_{i \rightarrow j} \beta \frac{r_i^{(t-1)}}{d_i}$

- $r'_j^{(t)} = \mathbf{0}$  if in-deg. of  $j$  is  $\mathbf{0}$

- **Now re-insert the leaked PageRank:**

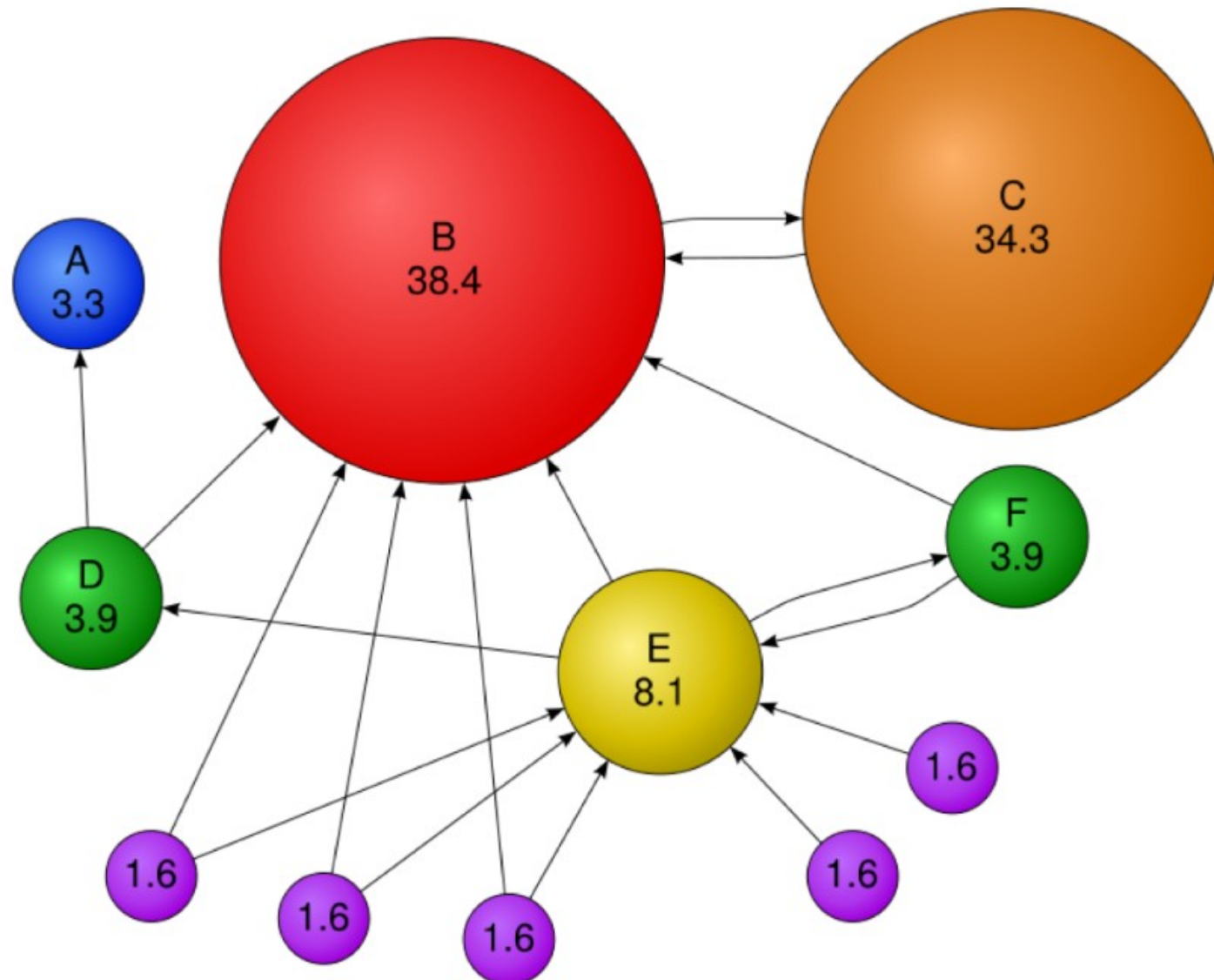
- $\forall j: r_j^{(t)} = r'_j^{(t)} + \boxed{\frac{1-S}{N}}$  where:  $S = \sum_j r'_j^{(t)}$

- $t = t + 1$

- **while**  $\sum_j |r_j^{(t)} - r_j^{(t-1)}| > \varepsilon$

# Example

Node size proportional to the PageRank score



# NetLogo: PageRank

network-choice  
Example 1

setup step go

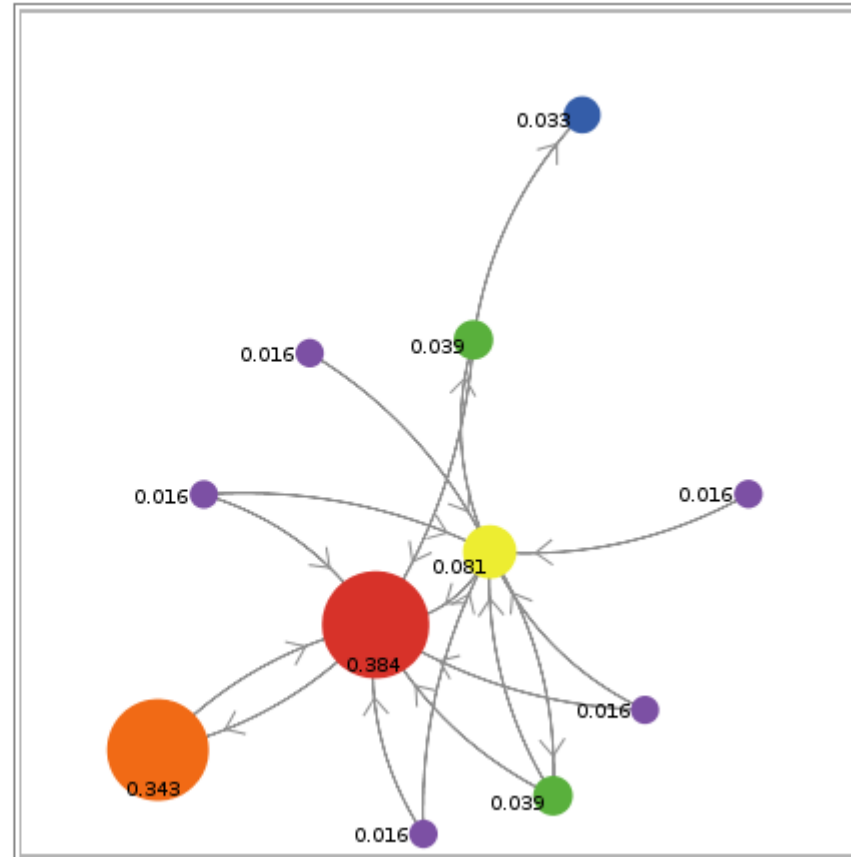
damping-factor 0.85

calculation-method  
diffusion

number-of-surfers 5

On  
 Off watch-surfers?

On  
 Off show-page-ranks?

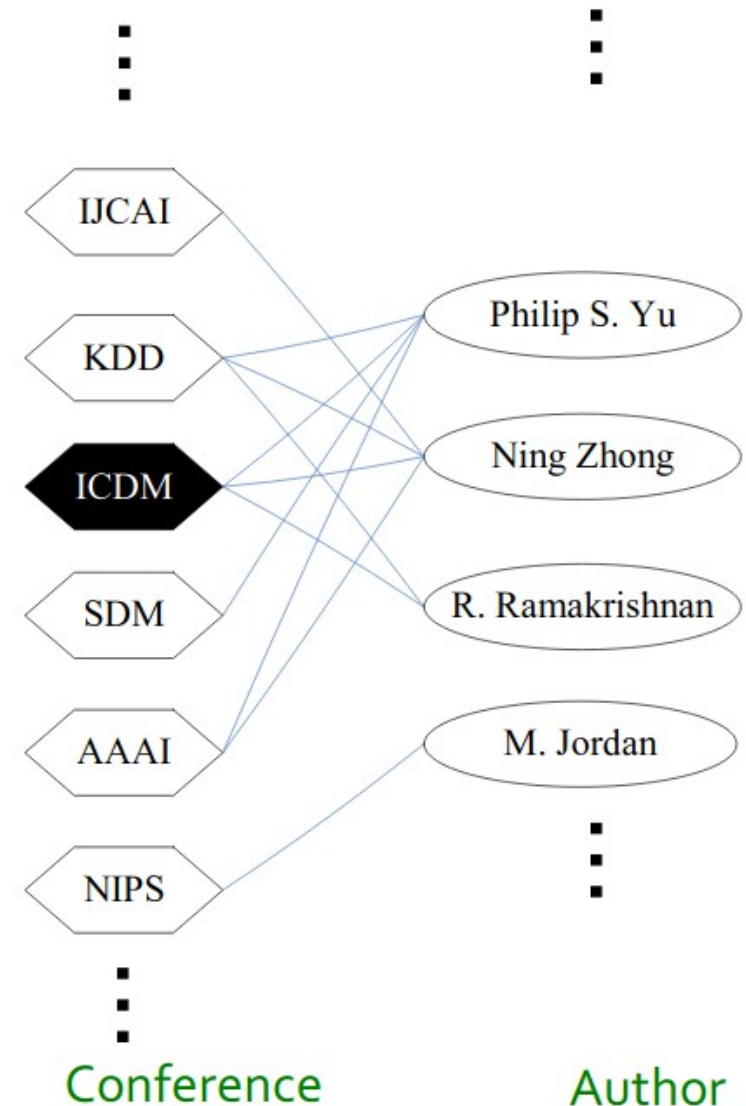


PageRank.nlogo

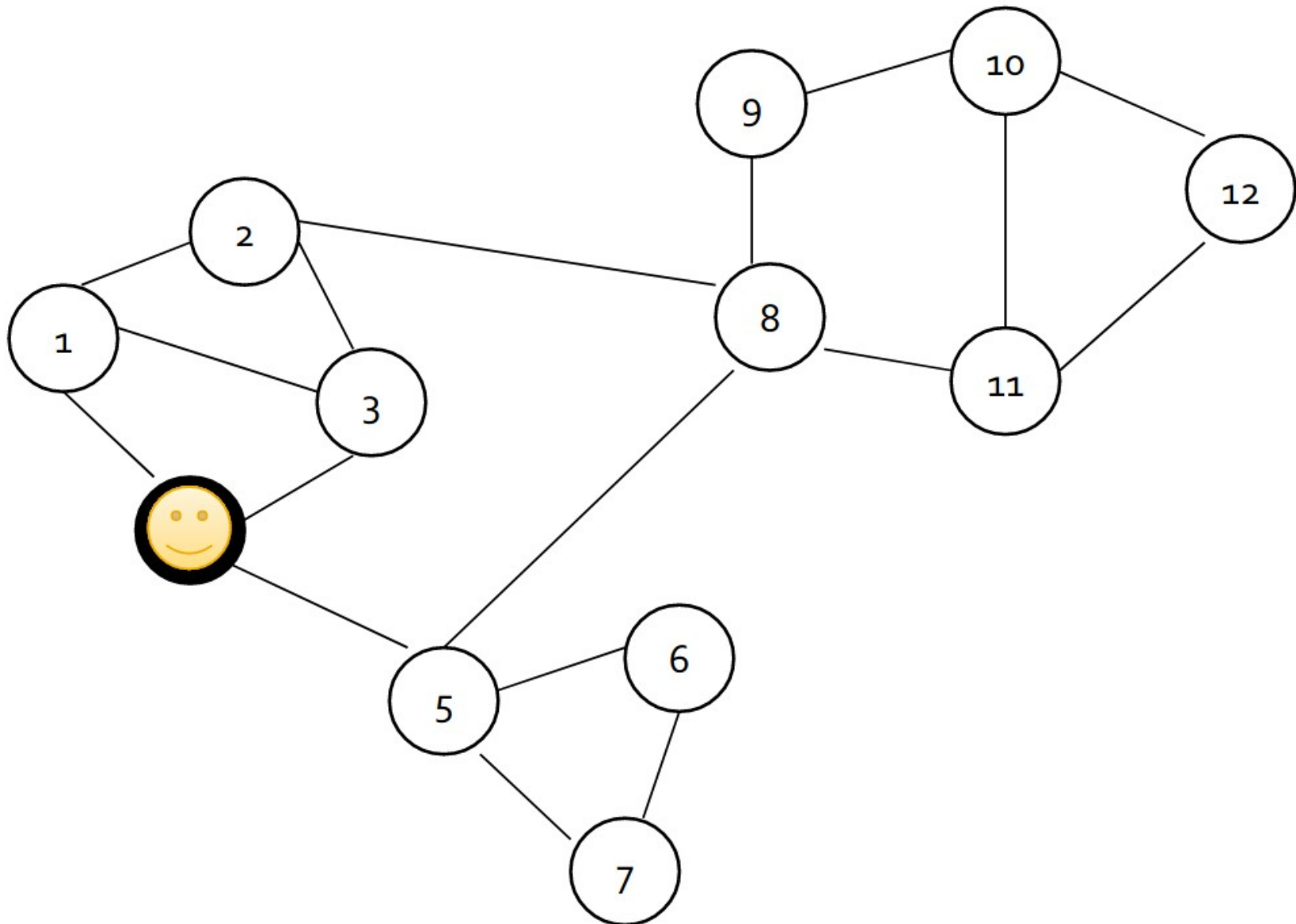
# Random Walk Restarts and Personalized PageRank

# Example Application: Graph Search

- **Given:**  
Conferences-to-authors graph
- **Goal:**  
Proximity on graphs
  - Q: What is most related conference to ICDM?



# Random Walk with Restarts

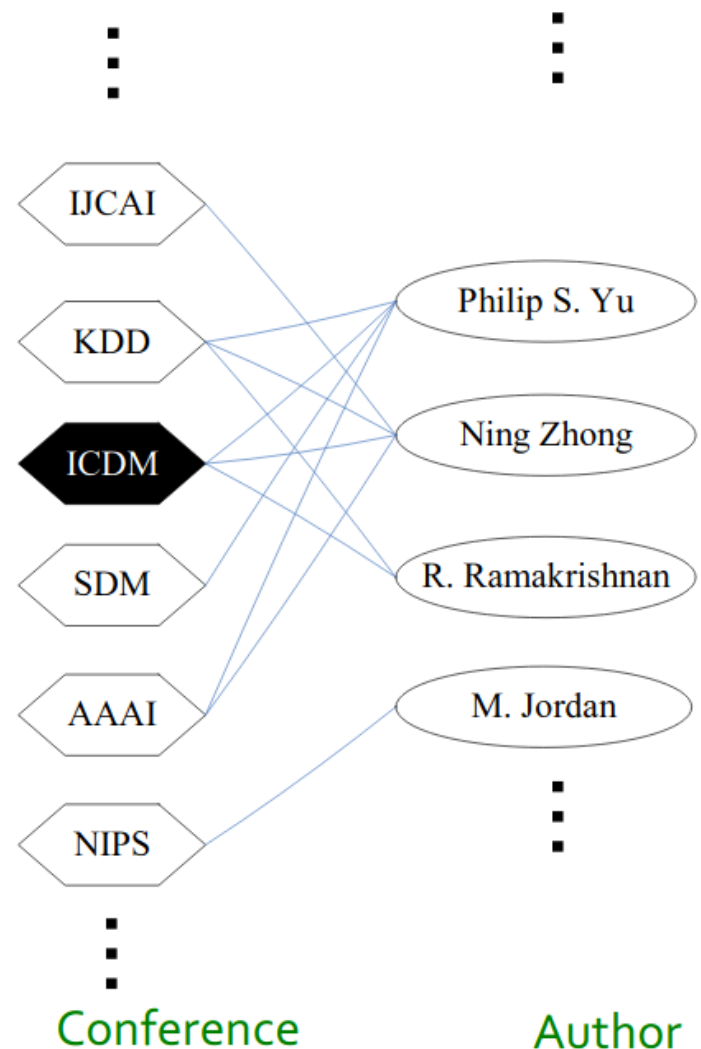


# Personalized PageRank

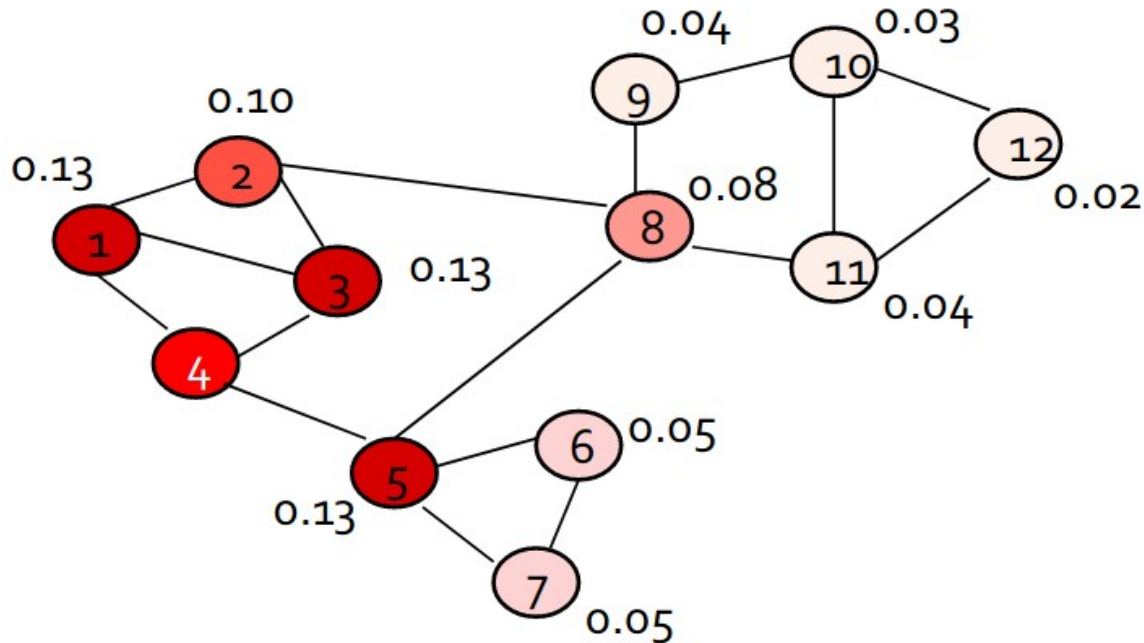
- **Goal:** Evaluate pages not just by popularity but by how close they are to the topic
- **Teleporting can go to:**
  - **Any page with equal probability**
    - PageRank (we used this so far)
  - **A topic-specific set of “relevant” pages**
    - Topic-specific (personalized) PageRank (**S ...teleport set**)  
$$M'_{ij} = \beta M_{ij} + (1 - \beta)/|S| \quad \text{if } i \in S$$
$$= \beta M_{ij} \quad \text{otherwise}$$
  - **A single page/node ( $|S| = 1$ ),**
    - Random Walk with Restarts

# PageRank: Applications

- **Graphs and web search:**
  - Ranks nodes by “importance”
- **Personalized PageRank:**
  - Ranks proximity of nodes to the teleport set  $S$
- **Proximity on graphs:**
  - **Q:** What is most related conference to **ICDM**?
  - **Random Walks with Restarts**
    - Teleport back to the starting node:  
 $S = \{ \text{single node} \}$



# Random Walk with Restarts



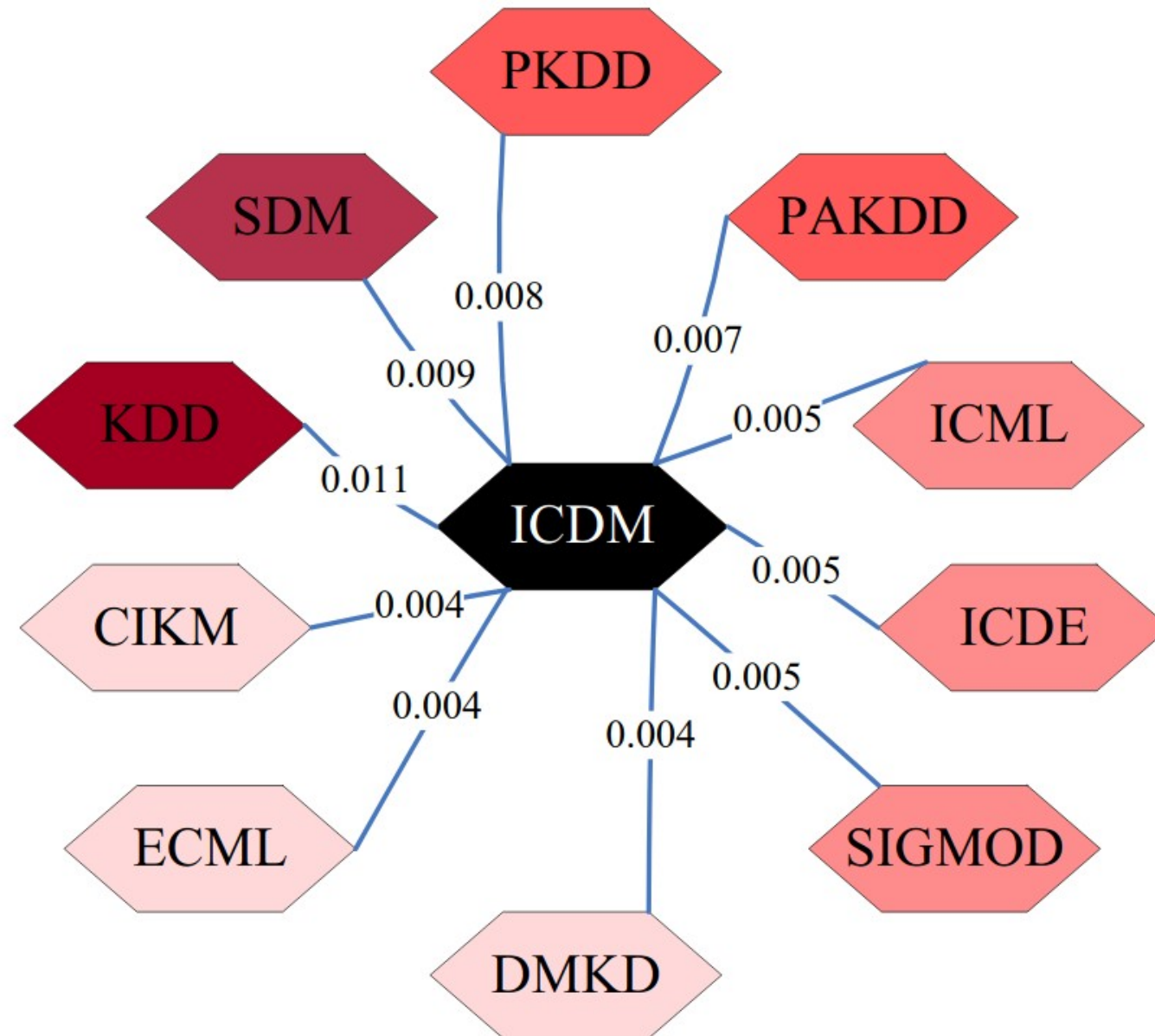
$S=\{4\}$

Notice: Nearby nodes have higher scores (are more red)

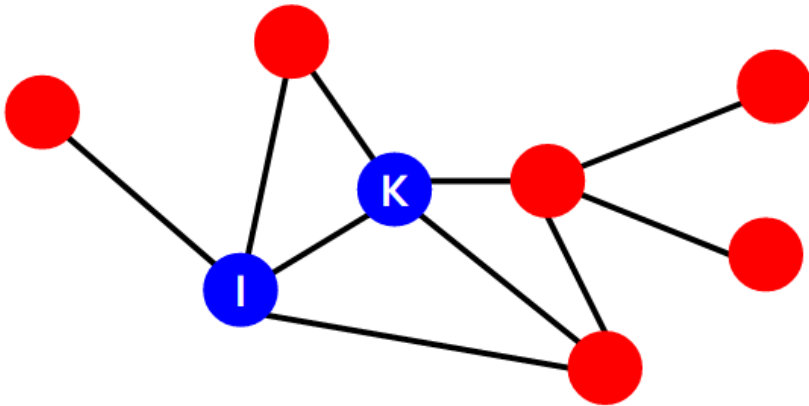
	Node 4
Node 1	0.13
Node 2	0.10
Node 3	0.13
Node 4	/
Node 5	0.13
Node 6	0.05
Node 7	0.05
Node 8	0.08
Node 9	0.04
Node 10	0.03
Node 11	0.04
Node 12	0.02

Ranking vector

# Most Related Conferences to ICDM



# Personalized PageRank



Graph of CS conferences

**Q:** Which conferences are closest to KDD & ICDM?

**A:** Personalized PageRank with teleport set  $S=\{\text{KDD}, \text{ICDM}\}$