HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Introduction to Data Science

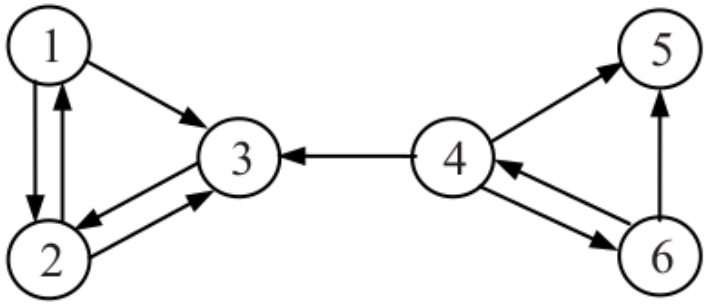
(IT4142E)

Contents

- Lecture 1: Overview of Data Science
- Lecture 2: Data crawling and preprocessing
- Lecture 3: Data cleaning and integration
- Lecture 4: Exploratory data analysis
- Lecture 5: Data visualization
- Lecture 6: Multivariate data visualization
- Lecture 7: Machine learning
- Lecture 8: Big data analysis
- Lecture 9: Capstone Project guidance
- **Lecture 10+11: Text, image, graph analysis**
- Lecture 12: Evaluation of analysis results

Links and hypertext

- Questions
  - Do the links represent authority to some pages? Is this useful for ranking?
  - How likely is a page, pointed to by the SOICT home page, about Math?



Social network

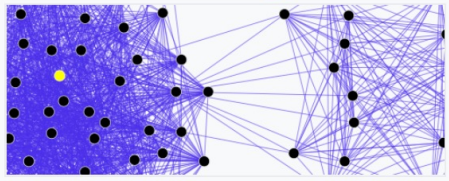
From Wikipedia, the free encyclopedia

This article is about the theoretical concept as used in the social and behavioral sciences. For social networking sites, see Social networking service. For the 2010 movie, see The Social Network. For other uses, see Social network (disambiguation).

A **social network** is a social structure made up of a set of social actors (such as individuals or organizations), sets of dyadic ties, and other social interactions between actors. The social network perspective provides a set of methods for analyzing the structure of whole social entities as well as a

Part of a series on

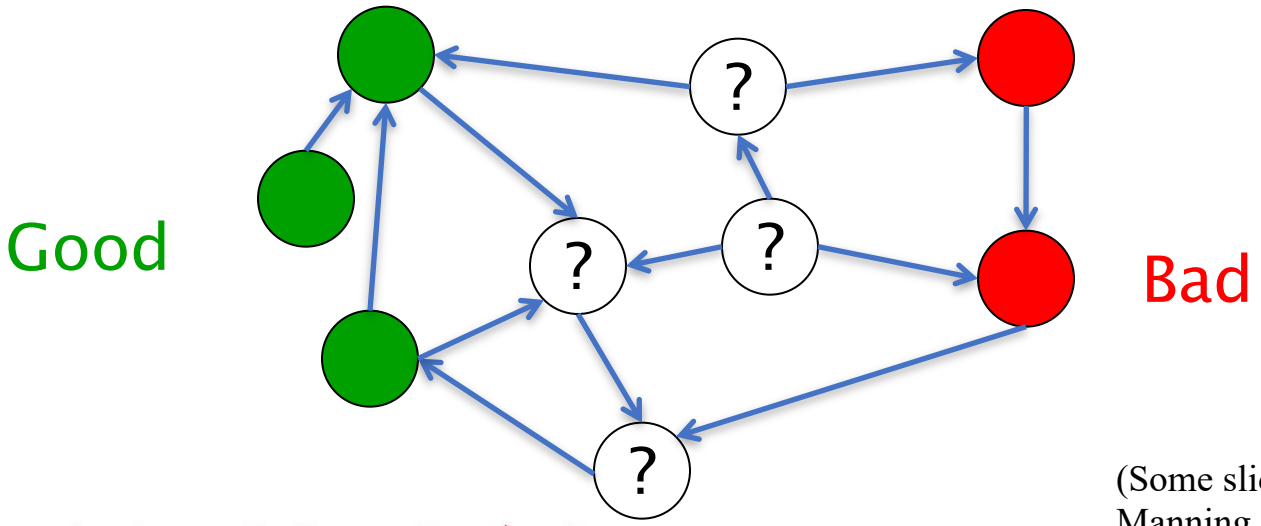
Sociology



- Application areas
  - The Web
  - Email
  - Social networks, …

Links are everywhere

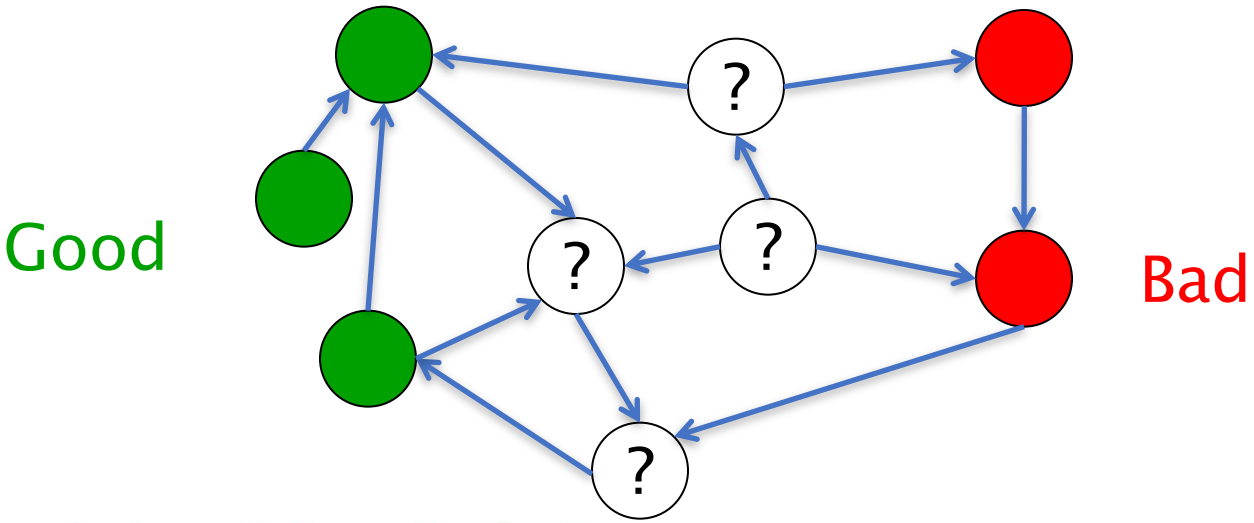
- Powerful sources of authenticity and authority
  - Mail spam – which email accounts are spammers?
  - Host quality – which hosts are "bad"?
  - Phone call logs, …
- The Good, The Bad and The Unknown



(Some slides came from Chris Manning and Pandu Nayak)

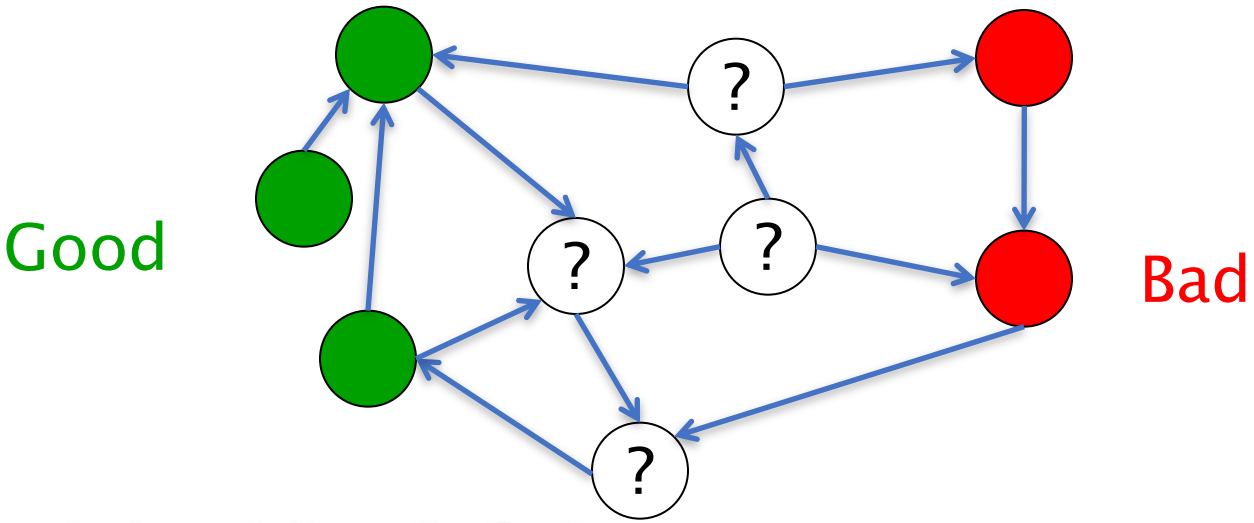
Example: Good/Bad/Unknown

- The Good, The Bad and The Unknown
  - Good nodes won't point to Bad nodes
  - All other combinations are plausible



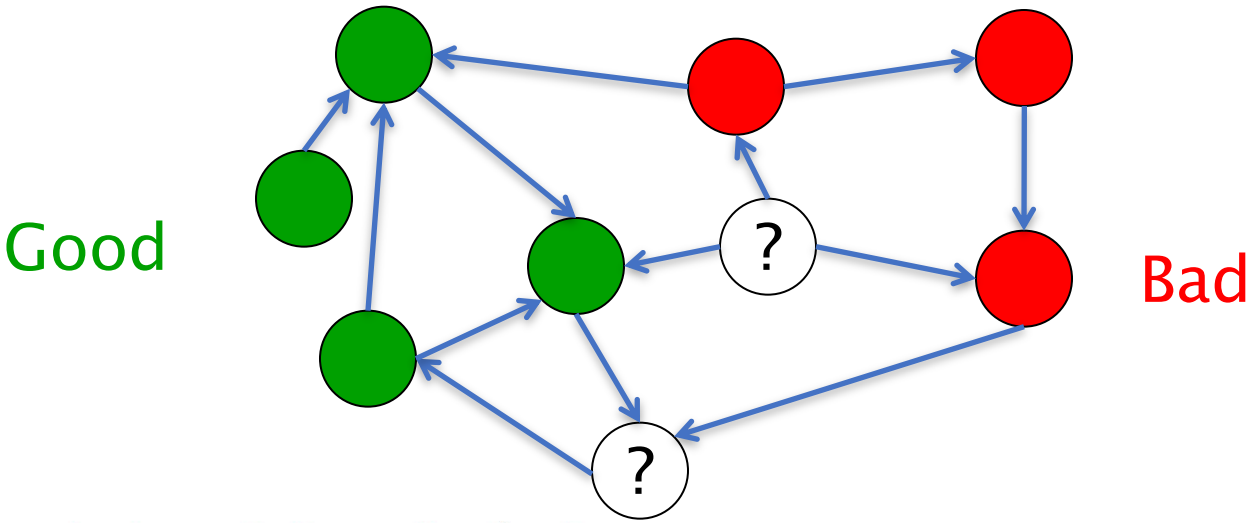
Simple iterative logic

- Good nodes won't point to Bad nodes
  - If you point to a Bad node, you're Bad
  - If a Good node points to you, you're Good



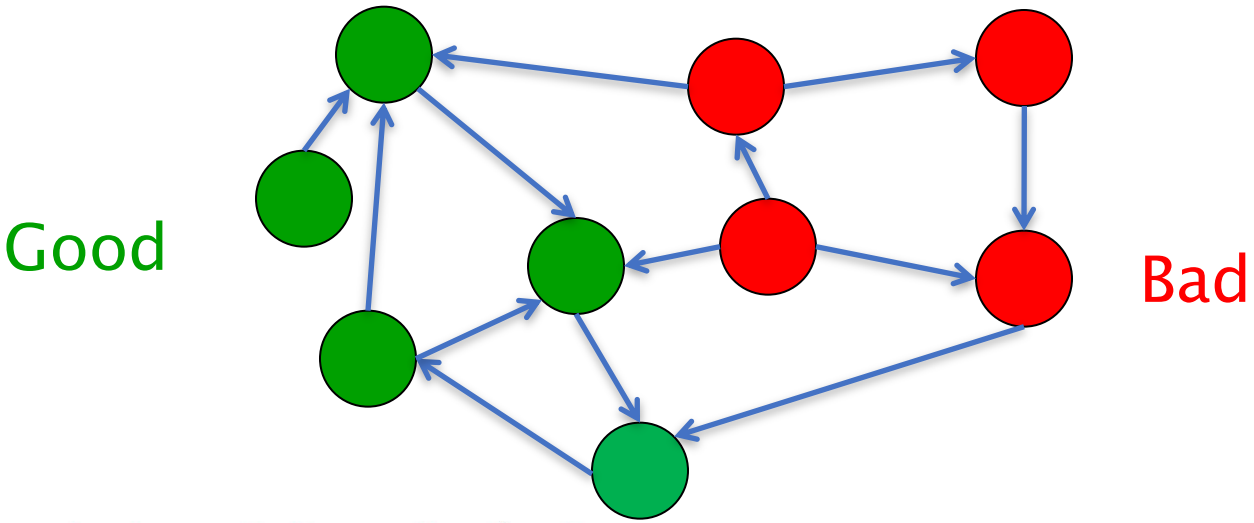
Simple iterative logic

- Good nodes won't point to Bad nodes
  - If you point to a Bad node, you're Bad
  - If a Good node points to you, you're Good



Simple iterative logic

- Good nodes won't point to Bad nodes
  - If you point to a Bad node, you're Bad
  - If a Good node points to you, you're Good



Many needs for link analysis

- Community detection in social networks
  - Detect user groups, each contains some users with similar behaviors/interest
- Shoppers' affinity
  - Consumers whose friends spend a lot, spend a lot themselves
- Citation analysis
  - Detect influential research from citation
- …

The reusable holdout: Preserving validity in adaptive data analysis

Cynthia Dwork,[1*] Vitaly Feldman,[2*] Moritz Hardt,[3*] Toniann Pitassi,[4*] Omer Reingold,[5*] Aaron Roth[6*]

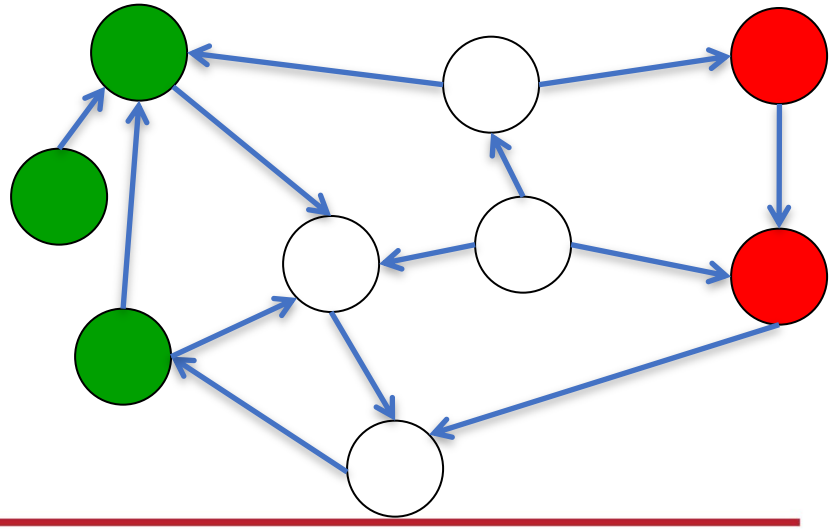
Misapplication of statistical data analysis is a common cause of spurious discoveries in scientific research. Existing approaches to ensuring the validity of inferences drawn from data assume a fixed procedure to be performed, selected before the data are examined. In common practice, however, data analysis is an intrinsically adaptive process, with new analyses generated on the basis of data exploration, as well as the results of previous analyses on the same data. We demonstrate a new approach for addressing the challenges of adaptivity based on insights from privacy-preserving data analysis. As an application, we show how to safely reuse a holdout data set many times to validate the results of adaptively chosen analyses.

Throughout the scientific community there is a growing recognition that claims of statistical significance in published research are frequently invalid. There has been a great deal of effort to understand and propose mitigations for this problem, largely focusing on statistical methods for controlling the false discovery rate in multiple hypothesis testing (*1*). However, the statistical inference theory surrounding this body of work assumes that a fixed procedure of these methods focus on a single round of adaptivity—such as variable selection followed by regression on selected variables or model selection followed by testing—and are optimized for specific inference procedures [the literature is too vast to adequately cover here, but see chapter 7 in (*5*) for a starting point]. There are also procedures for controlling false discovery in a sequential setting where tests arrive one-by-one (*6*–*8*). However, these results crucially depend on

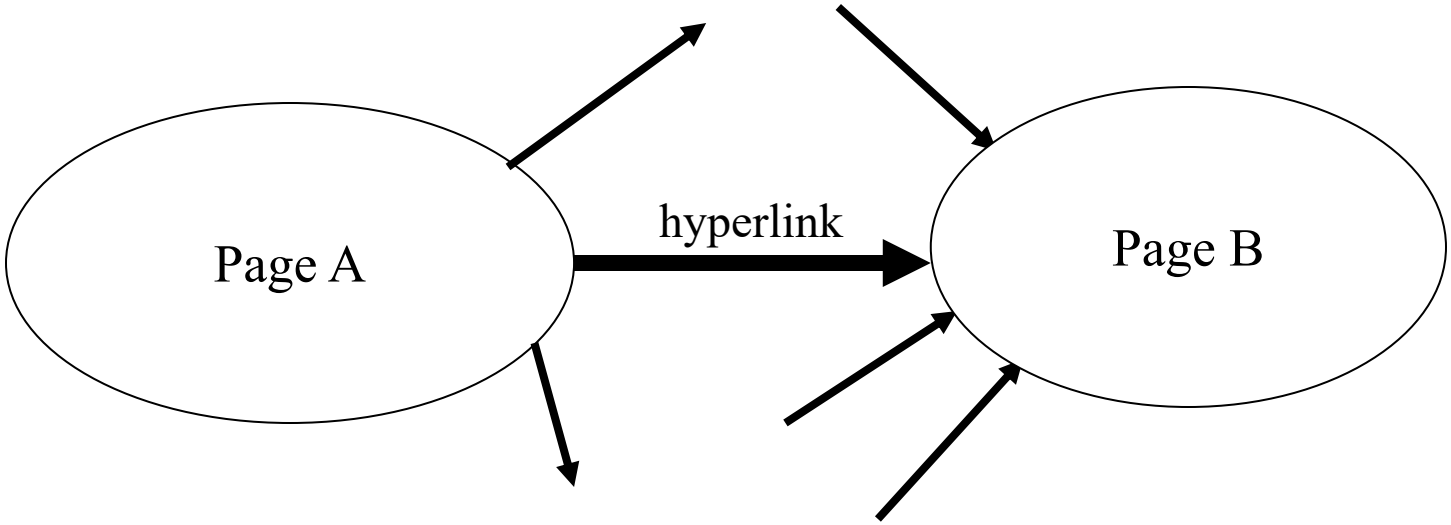
Cite (refer to) other research papers

Links and graphs

- **Vertex (node):** an entity of interest
  - E.g.: a user, a document, a web page, an organization, …
- **Edge:** the (directed) link from one vertex to one another
- **Graph:** $G = (V, E)$
  - V: a set of nodes
  - E: a set of edges that connect some nodes in V



The Web as a Directed Graph



Hypothesis: A hyperlink between pages denotes a conferral of authority (quality signal)

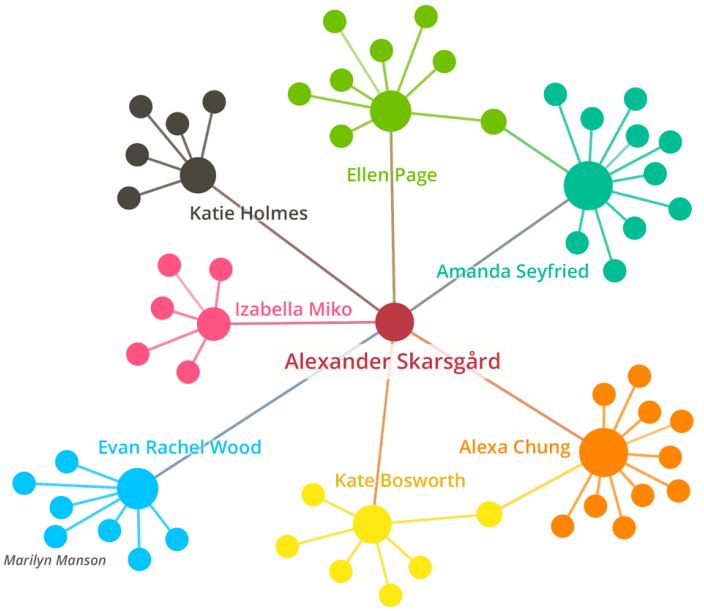
Popular tasks in Link Analysis

- Vertex ranking
- Community detection
- Node classification
- Link prediction

Centrality analysis

Centrality

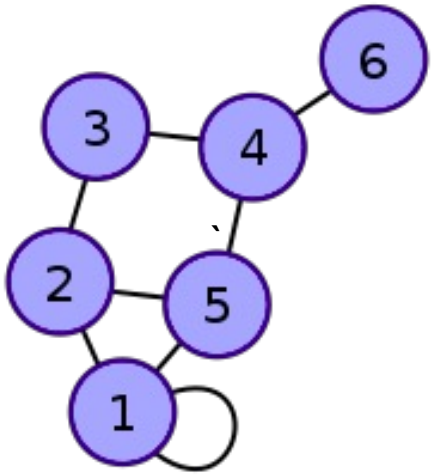
- What are important vertices?
  - Need a **measure**
- What characterizes an important vertex?
  - Centrality
- Applications:
  - Identify the most influential person(s) in a social network
  - Identify key infrastructure nodes on the Internet or urban networks
  - Identify super-spreaders of disease
  - …



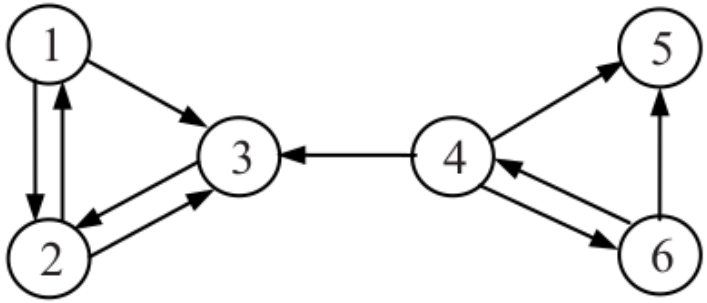
source: whosdatedwho.com

curiousgnu.com

Graph



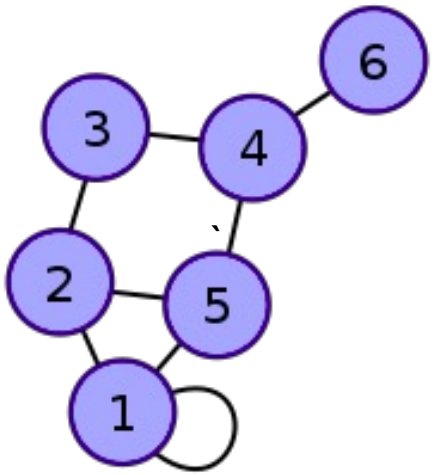
a) Undirected graph



b) Directed graph

Adjacency matrix

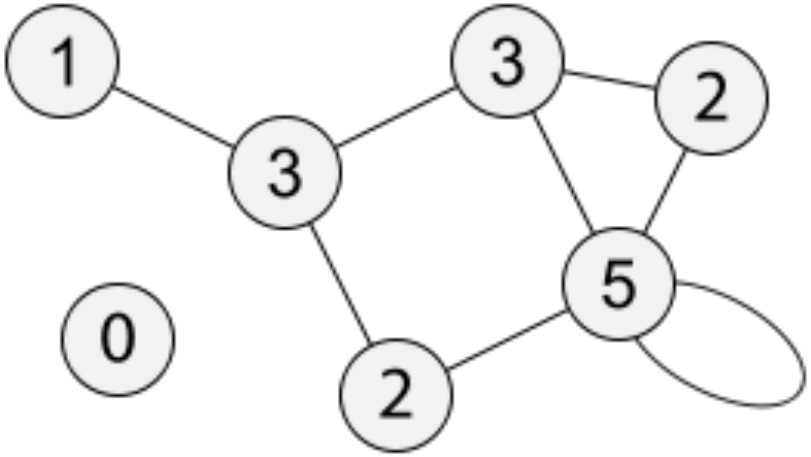
$$a[i,j] = \begin{cases} 1 & \text{if edge (i,j) exists} \\ 2 & \text{if } i = j \text{ and there exists a loop at node i} \\ 0 & \text{otherwise} \end{cases}$$

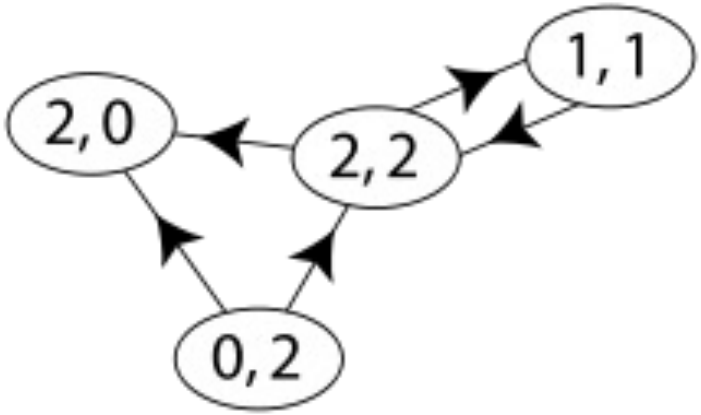


$$\begin{pmatrix} 2 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Vertex degree

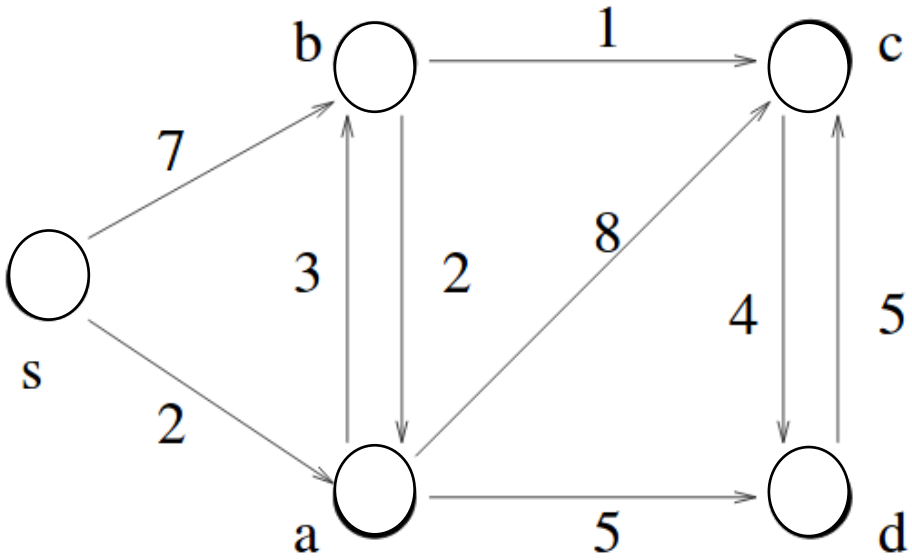
- $d_i(i)$ = number of edges to i
- $d_o(i)$ = number of edges from i





Weighted graph

- Each edge has a weight
- The whole graph can be represented by a weight matrix A
  - $a(i,j) = 0$ means no edge from node i to node j
  - $a(i,j) \neq a(j,i)$ sometimes



Dijkstra algorithm

- Find shortest path from *s* to other vertices
- *d(v)*: Distance from *s* to *v*

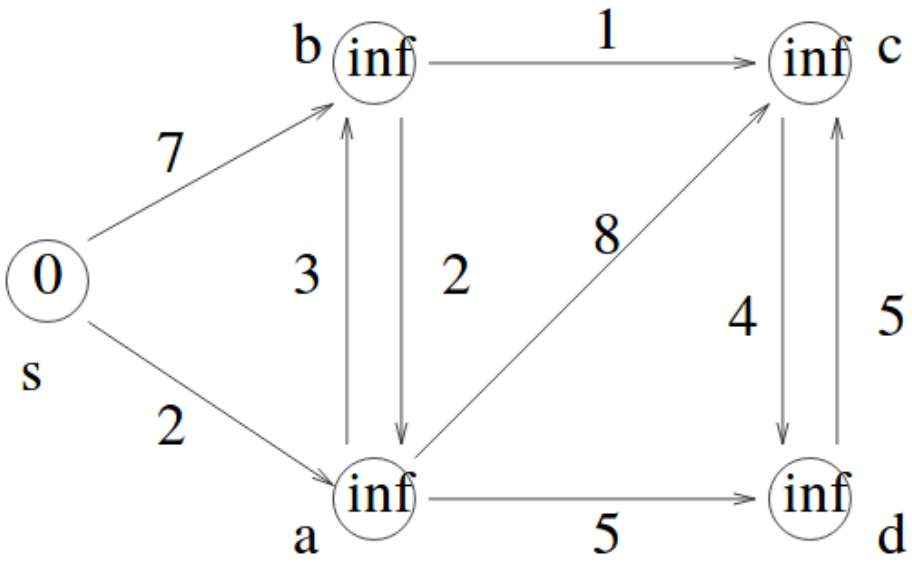
  1. *d(s) = 0; d(v) = ∞ for all vertices v ≠ s*

  2. *Enqueue all vertices v to priority queue Q*

  3. *Dequeue u from Q and update all d(v) (if necessary) for each v adjacent to u*

  *Return to step **2** until Q is empty*

Example

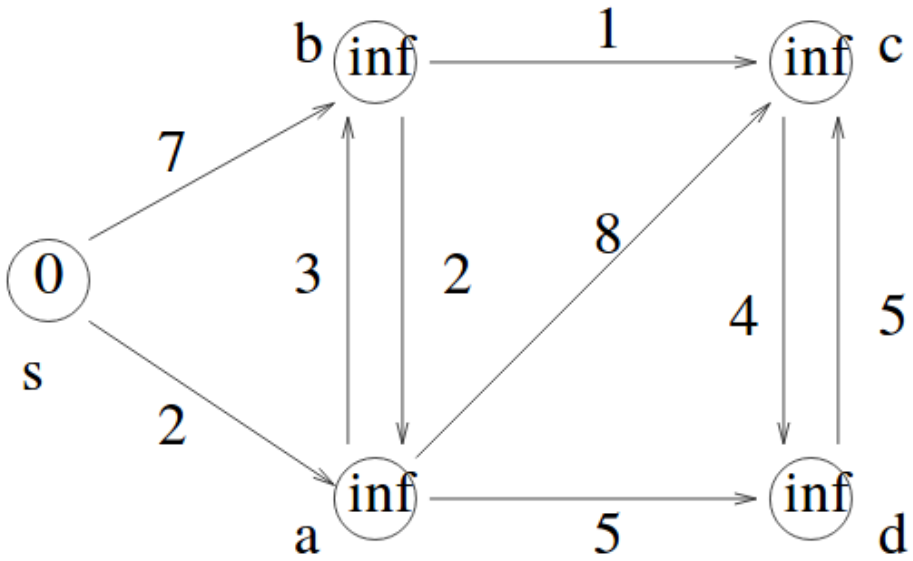


Example (cont)

v	s	a	b	c	d
$d[v]$	0	∞	∞	∞	∞
$pred[v]$	nil	nil	nil	nil	nil
$color[v]$	W	W	W	W	W

v	s	a	b	c	d
$d[v]$	0	∞	∞	∞	∞

Q

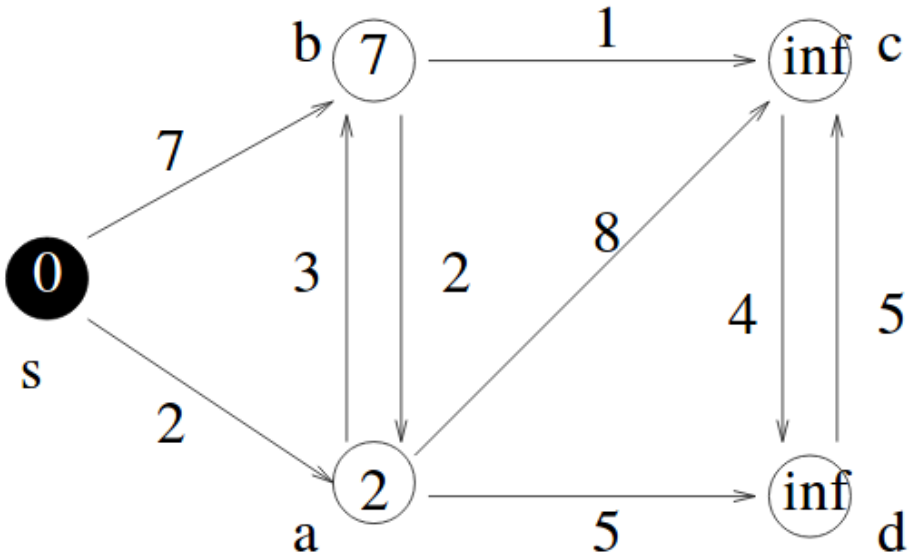


Example (cont)

v	s	a	b	c	d
$d[v]$	0	2	7	∞	∞
$pred[v]$	nil	s	s	nil	nil
$color[v]$	B	W	W	W	W

v	a	b	c	d
$d[v]$	2	7	∞	∞

Q

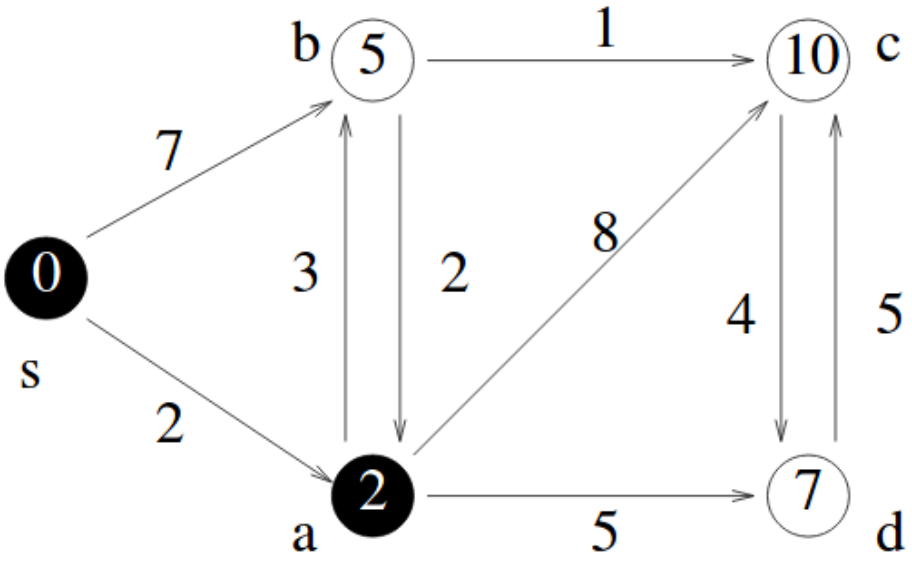


Example (cont)

v	s	a	b	c	d
$d[v]$	0	2	5	10	7
$pred[v]$	nil	s	a	a	a
$color[v]$	B	B	W	W	W

v	b	c	d
$d[v]$	5	10	7

Q

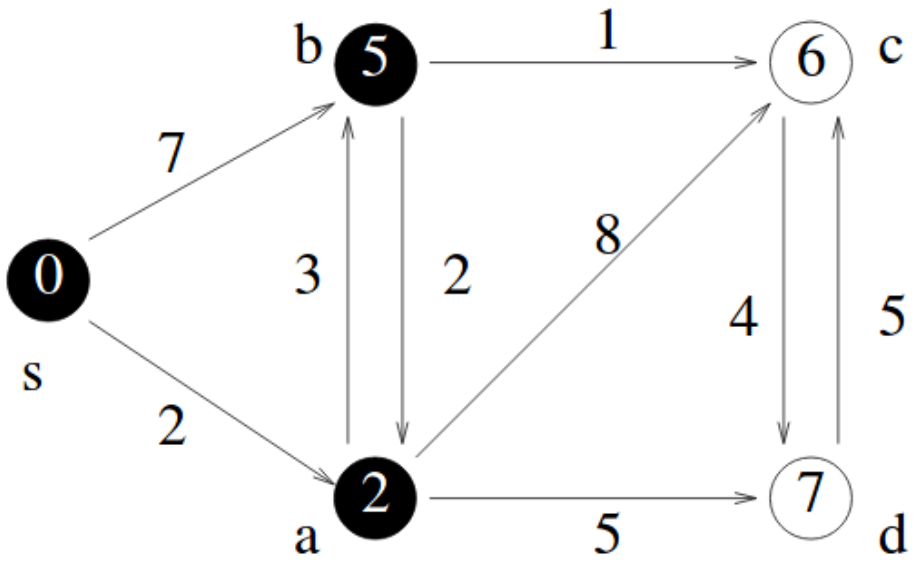


Example (cont)

v	s	a	b	c	d
$d[v]$	0	2	5	6	7
$pred[v]$	nil	s	a	b	a
$color[v]$	B	B	B	W	W

v	c	d
$d[v]$	6	7

Q

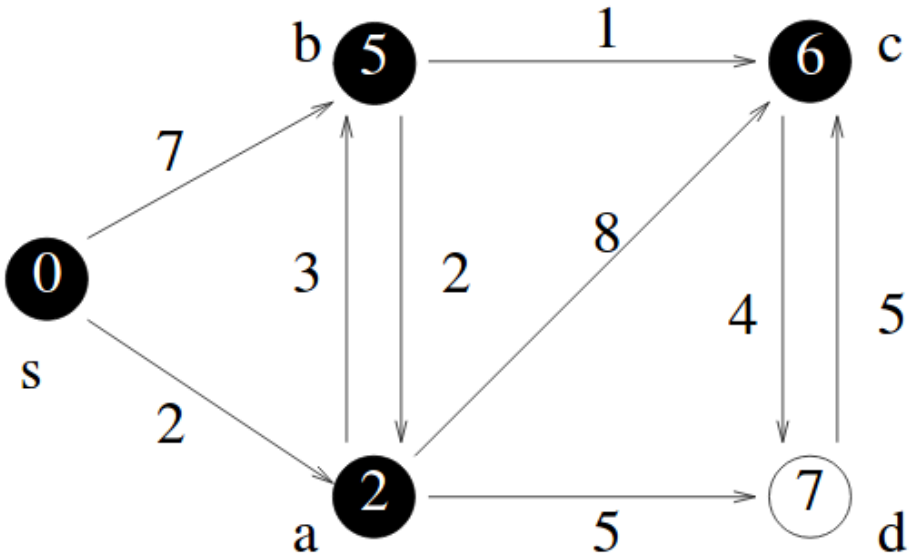


Example (cont)

v	s	a	b	c	d
$d[v]$	0	2	5	6	7
$pred[v]$	nil	s	a	b	a
$color[v]$	B	B	B	B	W

v	d
$d[v]$	7

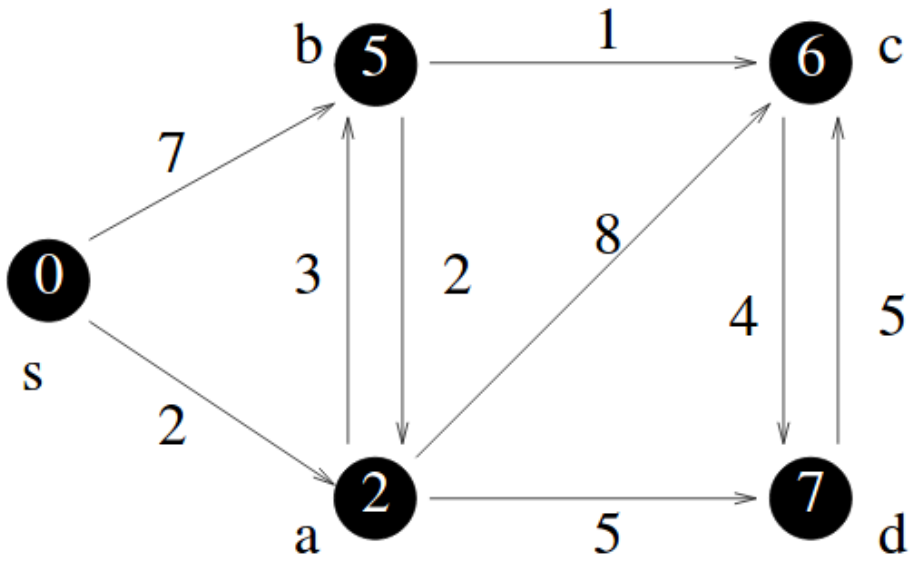
Q



Example (cont)

v	s	a	b	c	d
$d[v]$	0	2	5	6	7
$pred[v]$	nil	s	a	b	a
$color[v]$	B	B	B	B	B

$Q = \emptyset$



Closeness centrality

- Once we know how to compute the *(shortest) distance* from node i to node j (e.g., by using Dijkstra algorithm)
  - d(i,j): shortest distance from i to j
  - n: the total number of nodes in our graph
- We can measure the centrality of node i by

$$C_C(i) = \frac{n-1}{\sum_{j=1}^{n} d(i,j)}$$

  - reciprocal of the farness
  - inverse of the average of distance from node i to any other nodes

Betweenness centrality

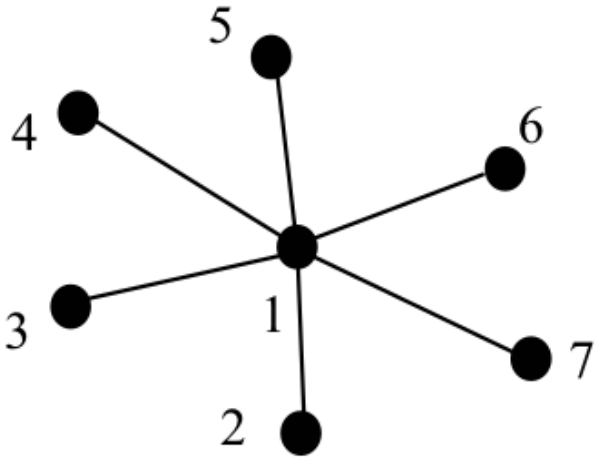
- Betweenness centrality of node i is defined by

$$C_B(i) = \sum_{j \neq k \neq i} \frac{p_{jk}(i)}{p_{jk}}$$

  - $p_{jk}(i)$: number of shortest paths from j to k that pass i
  - p_{jk}: number of shortest paths from node j to node k

- Indicate the number of times a node acts as a *bridge* along the shortest path between two other nodes
  - Higher implies probably more important

$C_B(1) = 15,$
$C_B(2) = C_B(3) = C_B(4) = 0$
$C_B(5) = C_B(6) = C_B(7) = 0$



Degree prestige

- Degree prestige use the degree of a node to see importance

$$P_D(j) = \frac{d_i(j)}{n-1}$$

  - $d_i(j)$: number of edges to node j
  - Higher implies probably more important

Proximity prestige

$$P_P(i) = \frac{1}{n-1} \frac{|I_i|}{\sum_{j \in I_i} \frac{d(i,j)}{|I_i|}}$$

- I_i: set of vertices that could reach i
- $|I_i|$: number of elements in I_i

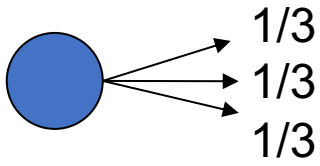
PageRank for ranking

PageRank

- PageRank was developed by Larry Page and Sergey Brin in 1996,
  - (Probably) One main component of Google search engine
- Ranking is based on the whole structure of the graph
  - Popularity of a web site hides in how many other sites had linked to it
- For large graph, ranking is approximated by iterative '*random walk*'

Pagerank scoring

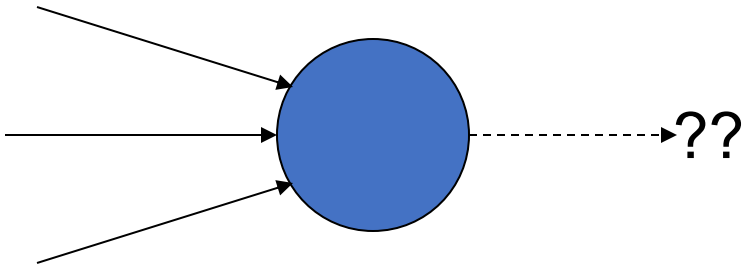
- Imagine a user doing a random walk on web pages:
  - Start at a random page
  - At each step, go out of the current page along one of the links on that page, equiprobably



- "In the long run" each page has a long-term visit rate → use this as the page's score.

Not quite enough

- The web is full of dead-ends.
  - Random walk can get stuck in dead-ends.
  - Makes no sense to talk about long-term visit rates.



Teleporting

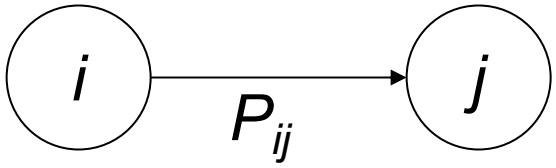
- At a dead end, jump to a random web page.
- At any non-dead end, with probability 10%, jump to a random web page.
  - With remaining probability (90%), go out on a random link.
  - 10% - a parameter.

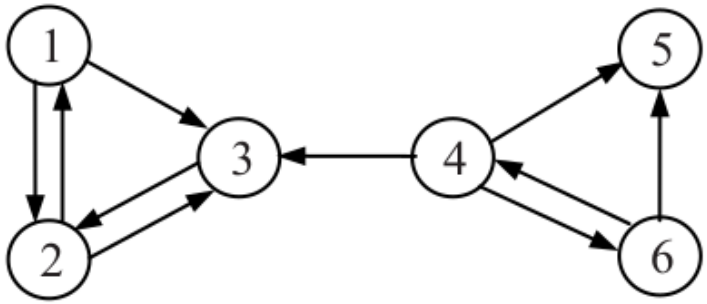
Result of teleporting

- Now cannot get stuck locally.
- There is a long-term rate at which any page is visited.
- How do we compute this visit rate?

Markov chains

- A Markov chain consists of n states, plus an $n \times n$ transition probability matrix **P**.
- At each step, we are in one of the states.
- For $1 \leq i,j \leq n$, the matrix entry P_{ij} tells us the probability of j being the next state, given that we are currently in state i.

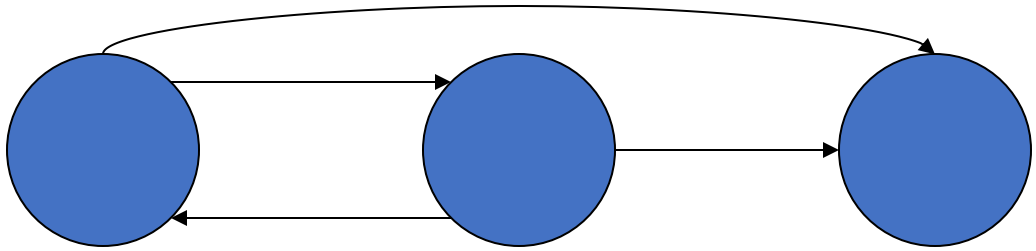




$$\mathbf{P} = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 0 & 1/3 & 1/3 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 1/2 & 0 \end{pmatrix}.$$

Markov chains

- Clearly, for all i, $\sum_{j=1}^{n} P_{ij} = 1$.
- Markov chains are abstractions of random walks.
- *Exercise*: represent the teleporting random walk from 3 slides ago as a Markov chain, for this case:



Ergodic Markov chains

- A Markov chain is called an *ergodic* or *irreducible* Markov chain if it is possible to eventually get from every state to every other state with positive probability.
- For any *ergodic* Markov chain, there is a unique long-term visit rate for each state.
  - *Steady-state probability distribution*.
- Over a long time-period, we visit each state in proportion to this rate.
- It does not matter where we start.

Probability vectors

- A probability (row) vector $\mathbf{x} = (x_1, \ldots x_n)$ tells us where the walk is at any point.
  - E.g., (000…1…000) means we're in state i.

1 i n

- More generally, the vector $\mathbf{x} = (x_1, \ldots x_n)$ means the walk is in state i with probability x_i.

$$\sum_{i=1}^{n} x_i = 1$$

Change in probability vector

- If the probability vector is $\mathbf{x} = (x_1, \ldots x_n)$ at this step, what is it at the next step?
- Recall that row *i* of the transition matrix **P** tells us where we go next from state *i*.
- So from **x**, our next state is distributed as **xP**
  - The one after that is $\mathbf{xP}^2$, then $\mathbf{xP}^3$, etc.
  - (Where) Does this converge?

How do we compute this vector?

- Let **a** = $(a_1, \ldots a_n)$ denote the row vector of steady-state probabilities.
- If our current position is described by **a**, then the next step is distributed as **aP**.
- But **a** is the steady state, so **a** = **aP**.
- Solving this matrix equation gives us **a**.
  - So **a** is the (left) eigenvector for **P**.
  - (Corresponds to the "principal" eigenvector of **P** with the largest eigenvalue.)
  - Transition probability matrices always have largest eigenvalue 1.

Smooth version

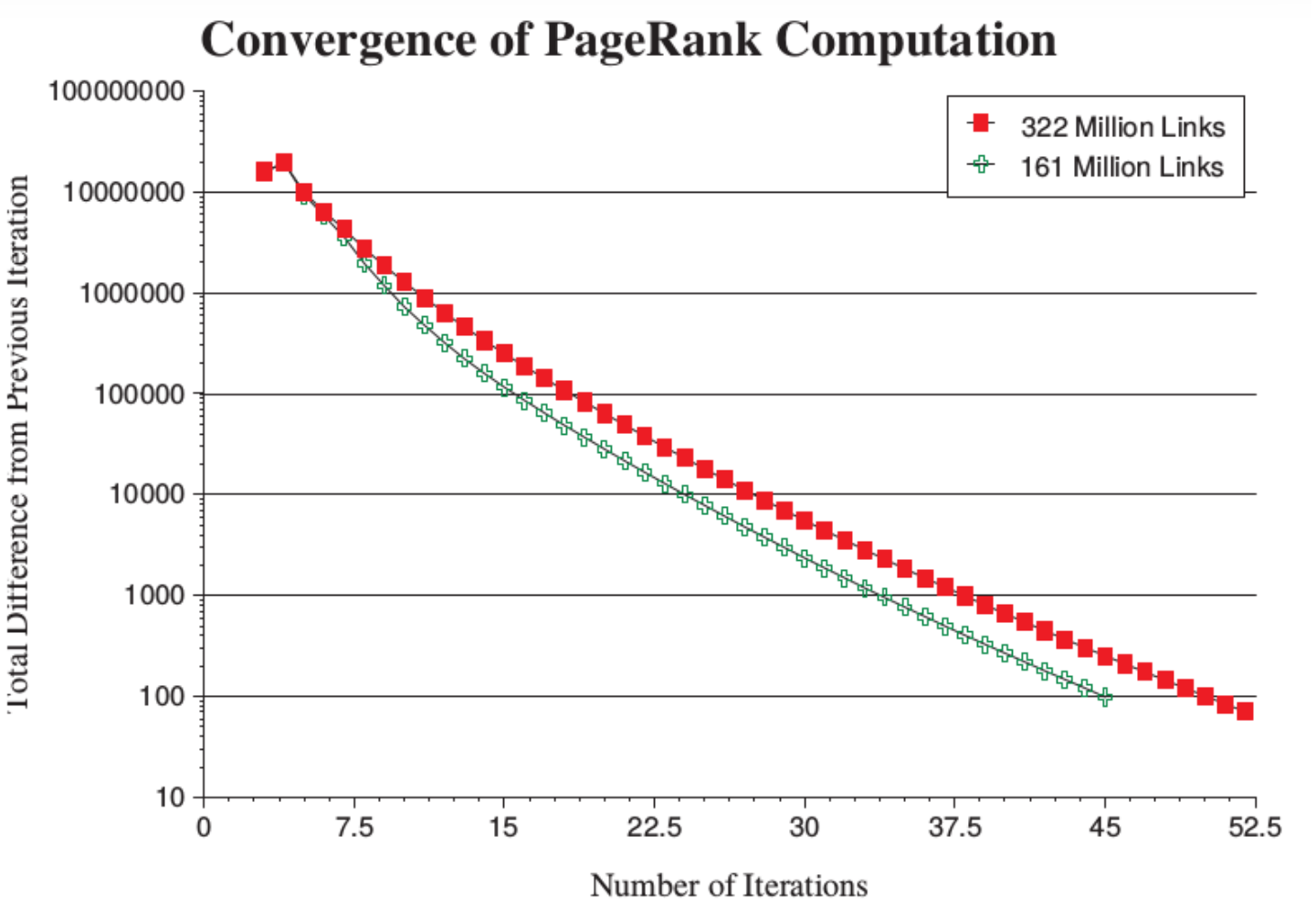
- To avoid zero columns and ensure ergodicity, replace **P** by:

$$\widehat{\boldsymbol{P}} = d\boldsymbol{P} + \frac{1-d}{n}\mathbf{1}$$

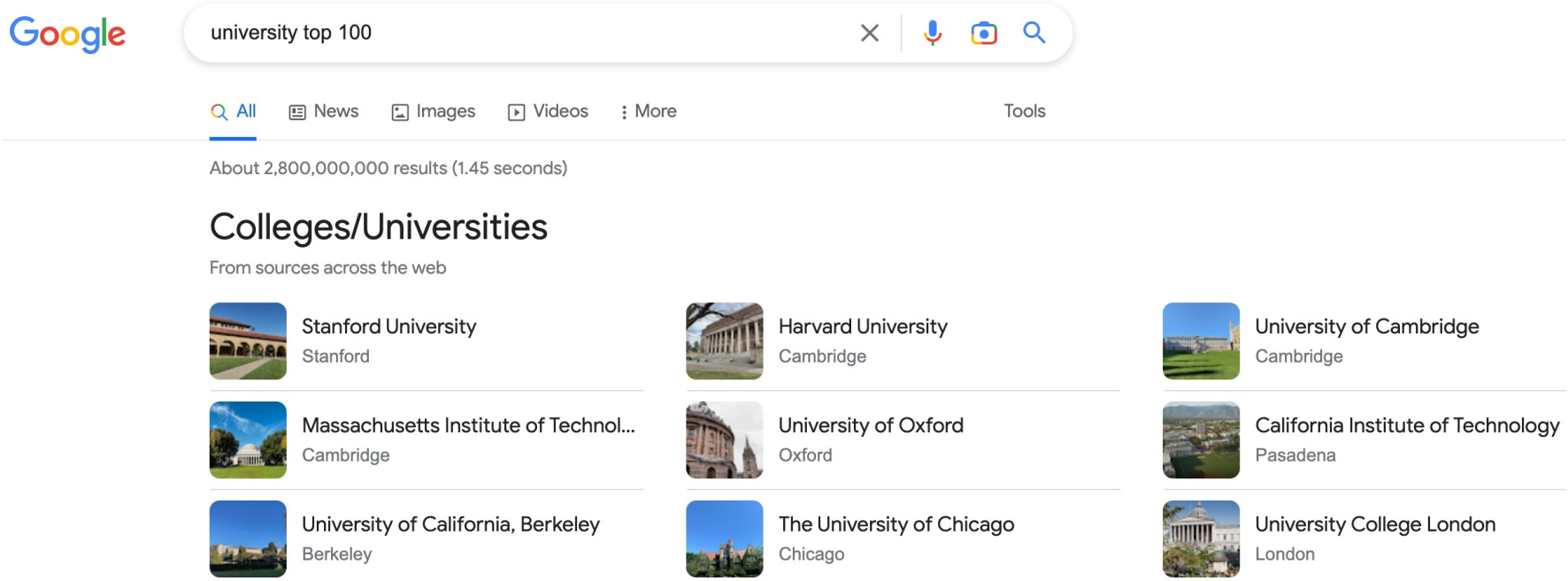
  - d is the damping factor, $d \in [0,1]$; $\mathbf{1}$ is the matrix of 1's

- Power method:
  - Initialize $\boldsymbol{x}^{(0)}$ (where we start)
  - At iteration t, compute $\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)}\widehat{\boldsymbol{P}}$
  - Until $\left\|\boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}\right\| < \epsilon$

Convergence

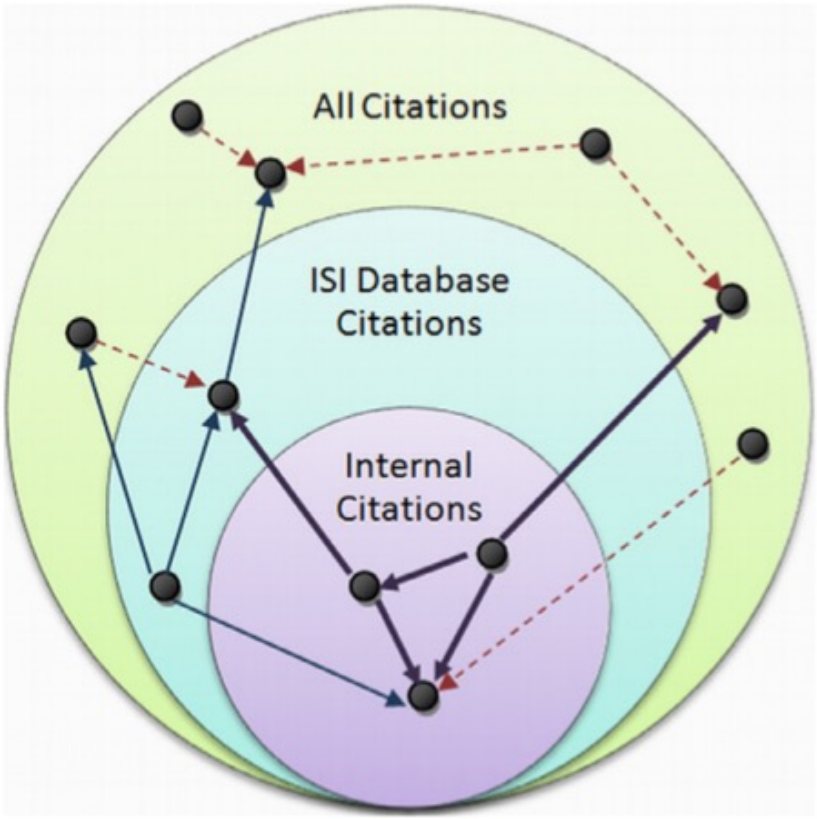


Application: Web search

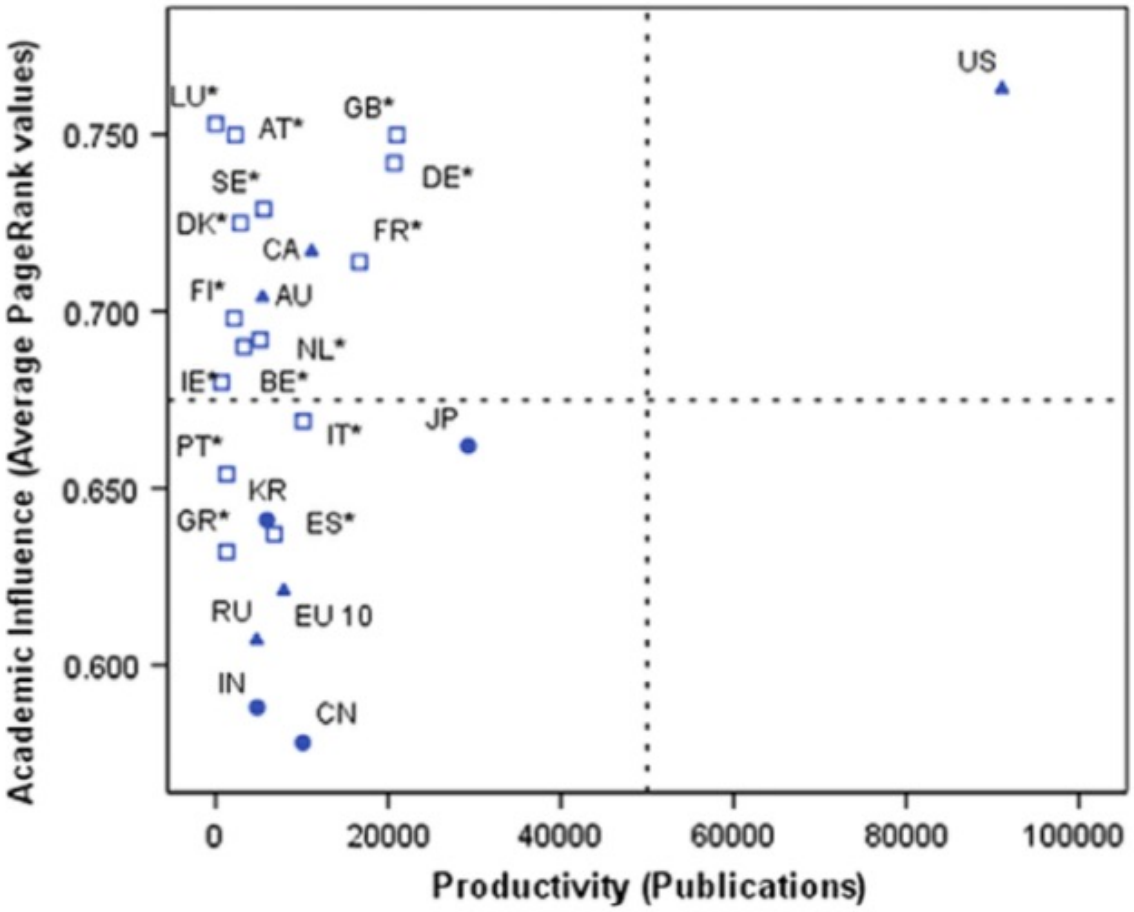


Application: Citation analysis

Guan et al. 2008. "*Bringing Page-Rank to the Citation Analysis*"



Application: Citation analysis (cont)



HITS
for ranking

HITS: Hypertext Induced Topic Search

Kleinberg, Jon M. "Authoritative sources in a hyperlinked environment." *Journal of the ACM (JACM)* 46, no. 5 (1999): 604-632.

	Spam filtering	Query relevance	Execution
HITS			Online
PageRank			Offline

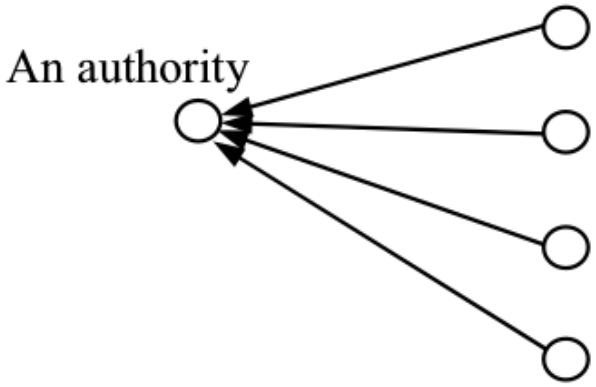
Hubs and Authorities

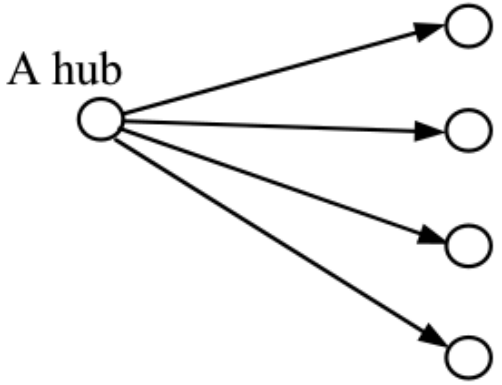
- A good hub page for a topic *points* to many authoritative pages for that topic.
- A good authority page for a topic is *pointed* to by many good hubs for that topic.
- Circular definition - will turn this into an iterative computation.

Hubs and Authorities

Authority: vertex with a large number of incoming edges

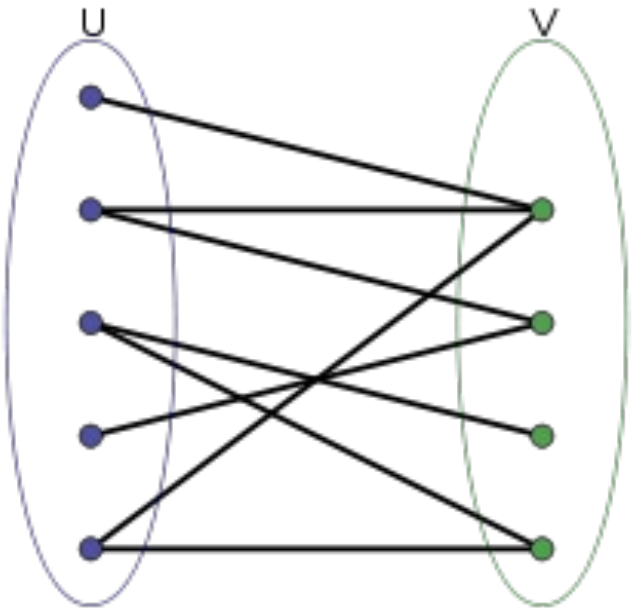
Hub: vertex with a large number of outgoing edges

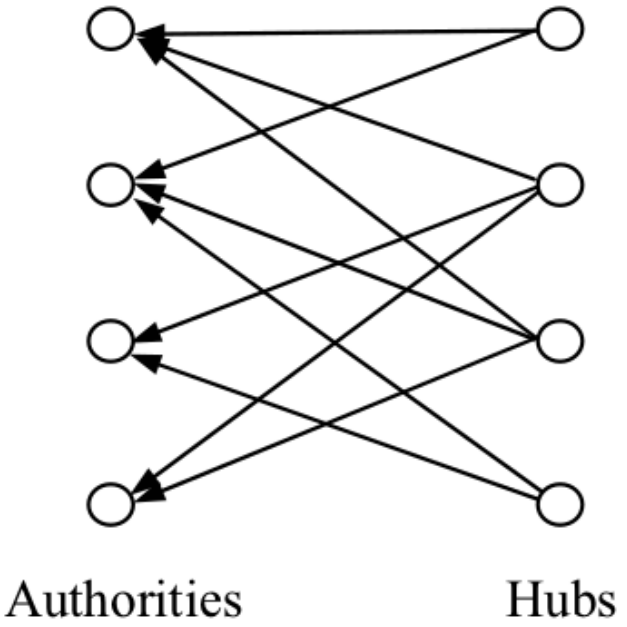




Bigraph

- The vertices is divided into two non-overlapped sets
- Each edge connects two vertices from two sets





HITS algorithm

Input: Query *q*

Output: Authority and hub score of relevant pages

Algorithm:

1. *Information retrieval*
2. *Graph expansion*
3. *Ranking*

1- Information retrieval

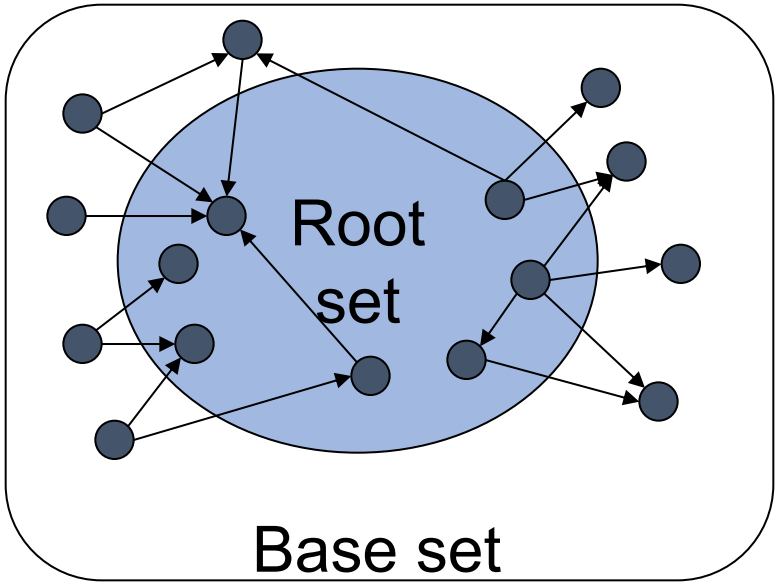
Use a search engine (Google, Bing)

- Create root **W** containing top *k* relevant pages of *q* (k = 200)

2- Graph expansion

From root **W**, expand to base **S**

- For each *p* in ***W***
  - Add pages to which *p* links
  - Add pages that link to *p*



3- Ranking

- From these, identify a small set of top hub and authority pages
  →iterative algorithm

Authority score (a)

Hub score (h)

Given a graph $G = (V, E)$, the scores can be computed as

$$\hat{a}(i) = \sum_{(j,i)\in E} h(j); \quad a(i) = \frac{\hat{a}(i)}{\|\hat{a}\|_1}$$

$$\hat{h}(i) = \sum_{(i,j)\in E} a(j); \quad h(i) = \frac{\hat{h}(i)}{\|\hat{h}\|_1}$$

3- Ranking (cont)

HITS-Iterate(G)

$\boldsymbol{a}_0 \leftarrow \boldsymbol{h}_0 \leftarrow (1, 1, \ldots, 1)$;

$k \leftarrow 1$

Repeat

$\boldsymbol{a}_k \leftarrow \boldsymbol{L}^T \boldsymbol{L}\boldsymbol{a}_{k-1}$;

$\boldsymbol{h}_k \leftarrow \boldsymbol{L}\boldsymbol{L}^T \boldsymbol{h}_{k-1}$;

$\boldsymbol{a}_k \leftarrow \boldsymbol{a}_k / ||\boldsymbol{a}_k||_1$; // normalization

$\boldsymbol{h}_k \leftarrow \boldsymbol{h}_k / ||\boldsymbol{h}_k||_1$; // normalization

$k \leftarrow k + 1$;

until $||\boldsymbol{a}_k - \boldsymbol{a}_{k-1}||_1 < \varepsilon_a$ and $||\boldsymbol{h}_k - \boldsymbol{h}_{k-1}||_1 < \varepsilon_h$;

return $\boldsymbol{a}_k$ and $\boldsymbol{h}_k$

$$\boldsymbol{a} = \boldsymbol{L}^T \boldsymbol{h}$$

$$\boldsymbol{h} = \boldsymbol{L}\boldsymbol{a}$$

$$\boldsymbol{L}_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY SOICT

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attentions!

soict.hust.edu.vn/ fb.com/groups/soict