

Marcel Gehrke

Data Understanding vs. Machine Training Community Analysis

Today's Lecture

- Social network analysis
- Anchor text
- Link analysis for ranking
 - PageRank and variants
 - Hyperlink-induced topic search (HITS)

Acknowledgements

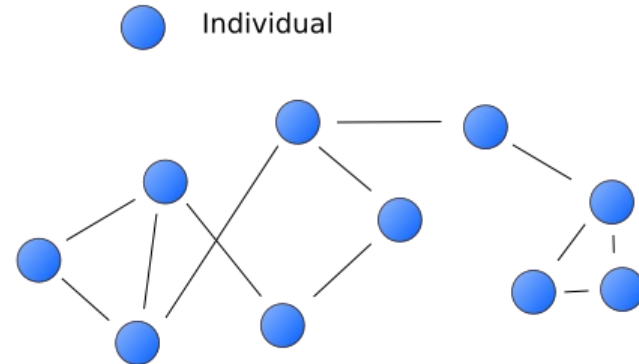
- Slides are based on material provided for
CS276A, Stanford Univ.,
Text Information Retrieval, Mining, and Exploitation
Chr. Manning, P. Raghavan, H. Schütze
- Thanks also to other lecturers who provided their teaching material on the web

Community Analysis

Social Network Analysis

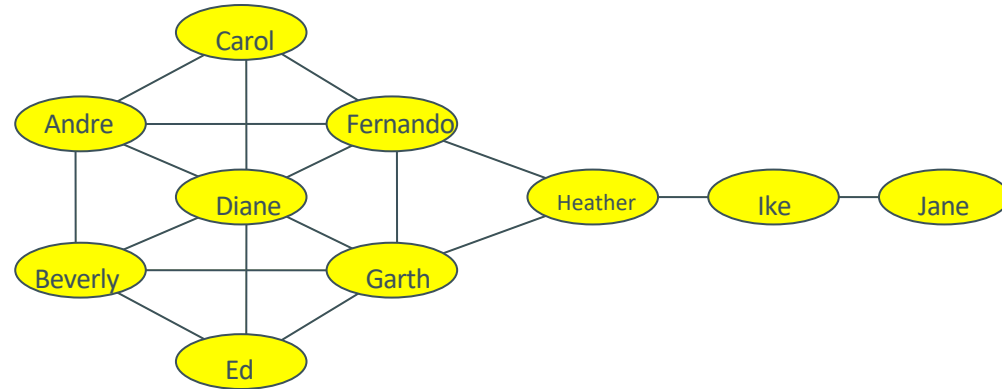
Social Network Analysis (SNA)

- Mapping and measuring of **relationships and flows** between people, groups, organizations, computers or other information/knowledge processing **entities**.
- The **nodes** in the network are the people and groups while the **links** show relationships or flows between the nodes.



Kite Network

- Who are **connectors** or **hubs** in the network?
- Who has **control** over what flows in the network?
- Who has best **visibility** of what is happening in the network?
- Who are **peripheral players**? Are they important?



Measures

1. Degree Centrality:

The number of direct connections a node has. What really matters is where those connections lead to and how they connect the otherwise unconnected.

$$C_D(n_i) = d(n_i), \quad C'_D(n_i) = \frac{d(n_i)}{g-1}$$

Rescaling by dividing through by the number of pairs of nodes not including n_i

2. Betweenness Centrality:

A node with high betweenness has great influence over what flows in the network indicating important links and single points of failure.

$$C_B(n_i) = \sum_{j < k} \frac{g_{jk}(n_i)}{g_{jk}}, \quad C'_B(n_i) = \frac{C_B(n_i)}{(g-1) \cdot (g-2)/2}$$



Undirected graph

3. Closeness Centrality:

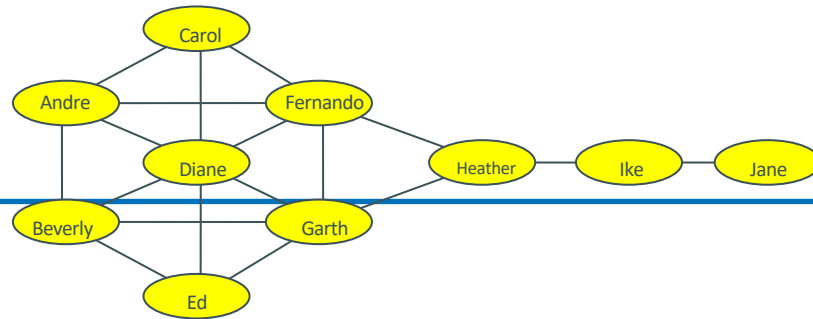
The measure of closeness of a node to everyone else. Determined by the sum of the length of the shortest paths between the node and all other nodes in the graph.

$$C_C(n_i) = \left[\sum_{j=1}^g d(n_i, n_j) \right]^{-1}, \quad C'_C(n_i) = \frac{g-1}{\sum_{j=1}^g d(n_i, n_j)} = (g-1) \cdot C_C(n_i)$$

Legend

- g = size of graph (number of nodes)
- $d(.)$ = (in)degree
- g_{jk} = number of minimal paths between nodes j and k
- $g_{jk}(n)$ = number of minimal paths between nodes j and k that contain n
- $(g - 1)(g - 2)/2$ = number of potential paths between two nodes
- $d(.,.)$ = distance between two nodes

- Scaling with $(g - 1)(g - 2)$: For every node n except n_i pair the node with all other nodes except n and n_i

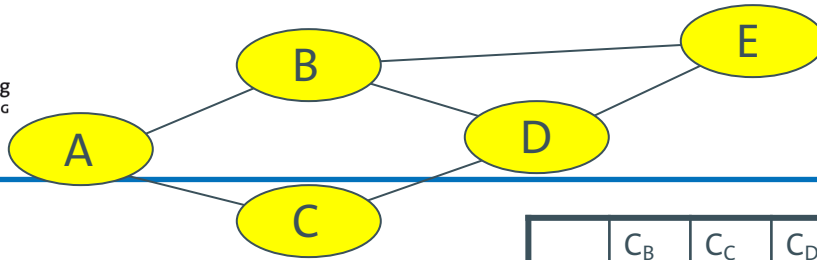


Example: Kite-Network

- $C_D(n_i) = d(n_i)$
- $C_B(n_i) = \sum_{j < k} \frac{g_{jk}(n_i)}{g_{jk}}$
- $C_C(n_i) = \left[\sum_{j=1}^g d(n_i, n_j) \right]^{-1}$

	C	A	F	D	B	G	E	H	I	J
C	0	1	1	1	0	0	0	0	0	0
A	1	0	1	1	1	0	0	0	0	0
F	1	1	0	1	0	1	0	1	0	0
D	1	1	1	0	1	1	1	0	0	0
B	0	1	0	1	0	1	1	0	0	0
G	0	0	1	1	1	0	1	1	0	0
E	0	0	0	1	1	1	0	0	0	0
H	0	0	1	0	0	1	0	0	1	0
I	0	0	0	0	0	0	0	1	0	1
J	0	0	0	0	0	0	0	0	1	0

	C_D
C	3
A	4
F	5
D	6
B	4
G	5
E	3
H	3
I	2
J	1



Example

- $C_D(n_i) = d(n_i)$
- $C_B(n_i) = \sum_{j < k} \frac{g_{jk}(n_i)}{g_{jk}}$
- $C_C(n_i) = \left[\sum_{j=1}^g d(n_i, n_j) \right]^{-1}$

	C_B	C_C	C_D
A	1	1/6	2
B	3	1/5	3
C	1	1/6	2
D	3	1/5	3
E	0	1/6	2

	A	B	C	D	E
A	0	1	1	0	0
B	1	0	0	1	1
C	1	0	0	1	0
D	0	1	1	0	1
E	0	1	0	1	0

Adjacency

	A	B	C	D	E
A	0	1	1	2	2
B	1	0	2	1	1
C	1	2	0	1	2
D	2	1	1	0	1
E	2	1	2	1	0

Distance

Marcel Gehrke

	A	B	C	D	E
A	0	A	A	BC	B
B	B	0	AD	B	B
C	C	AD	0	C	D
D	BC	D	D	0	D
E	B	E	D	E	0

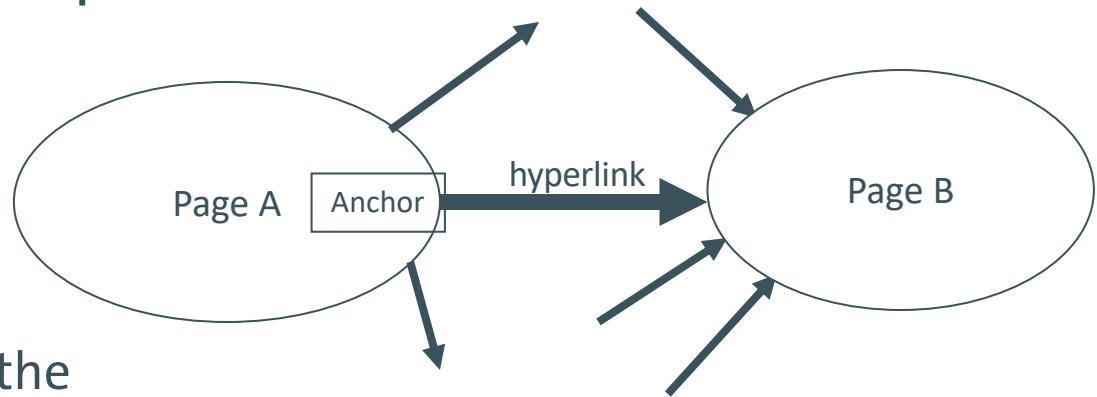
Paths

Community Analysis

Anchor Text

The Web as a Directed Graph

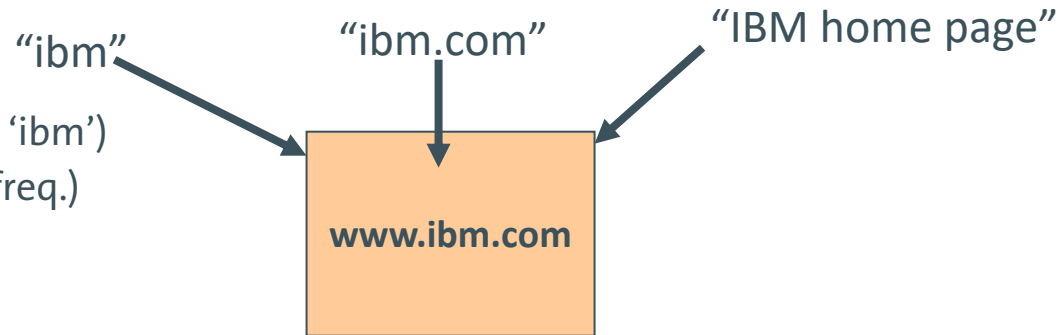
- **Assumption 1:** A hyperlink between pages denotes author perceived relevance (quality signal)
- **Assumption 2:** The anchor of the hyperlink describes the target page (textual context)



Anchor Text

- For IBM how to distinguish between:
 - IBM's home page (mostly graphical)
 - IBM's copyright page (high term freq. for 'ibm')
 - Rival's spam page (arbitrarily high term freq.)

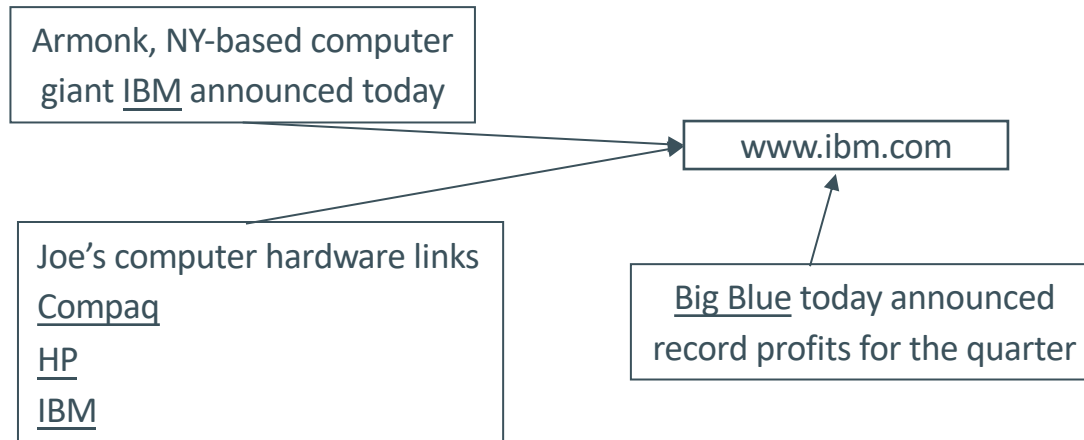
A million pieces of
anchor text with "ibm"
send a strong signal



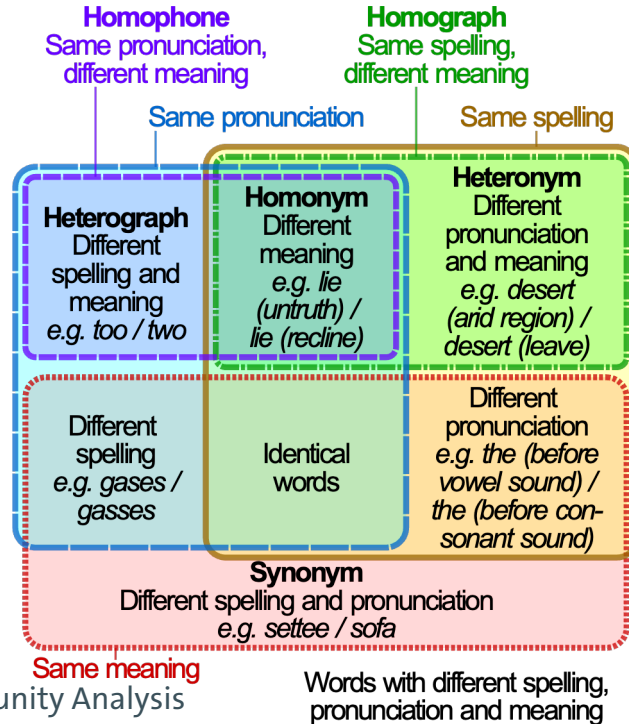
Oliver A. McBryan. GENVL and WWW: Tools for Taming the Web. Research explained at First International Conference on the World Wide Web. CERN, Geneva (Switzerland), May 25-26-27 1994 (WWW=World Wide Web Worm, first search engine for the web)

Indexing Anchor Text

- When indexing a document D , include anchor text from links pointing to D .



The Web as a Resource for NLP

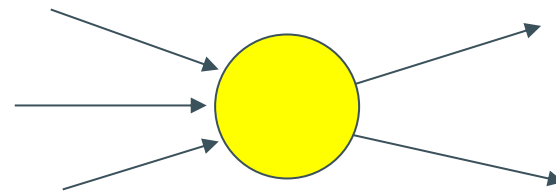


Community Analysis

PageRank


The Web as a Resource for Ranking

- First generation: using **link counts** as simple measures of **popularity**.
- Two basic suggestions:
 - Undirected popularity:
 - Each page gets a score = the number of in-links plus the number of out-links (3+2=5).
 - Directed popularity:
 - Score of a page = number of its in-links (3).



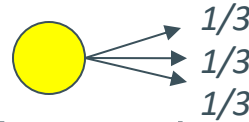
Query Processing

- First retrieve all pages matching the text query (say venture capital).
- Order these by their link popularity (either variant on the pre



How to organize for
"Search Engine
Optimization"?

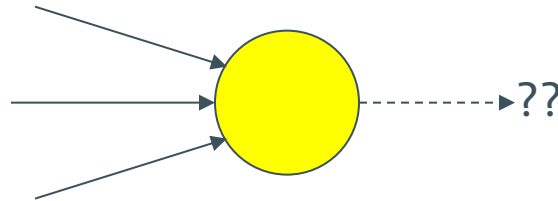
PageRank Scoring



- Imagine a browser 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
- 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.

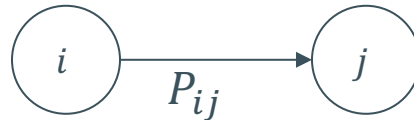
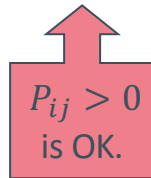


Teleporting / Damping

- 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.
- There is a long-term rate at which any page is visited.
 - How do we compute this visit rate?

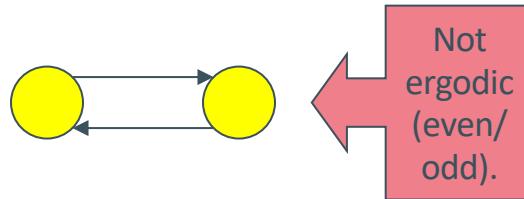
Markov Chains

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



Ergodic Markov Chains

- A Markov chain is ergodic if
 - you have a path from any state to any other (reducibility)
 - returns to states occur at irregular times (aperiodicity)
 - For any start state, after a finite transient time T_0 , the probability of being in any state at a fixed time $T > T_0$ is nonzero. (positive recurrence)



Ergodic Markov Chains

- For any ergodic Markov chain, there is a unique long-term visit rate for each state.
 - "Steady-state" distribution.
- Over a long time-period, we visit each state in proportion to this rate.
- It does not matter where we start.

State Vectors

- A (row) vector (state vector) $x = (x_1, \dots, x_n)$ tells us where the walk is at any point.
- E.g., $(\underset{1}{000}\dots\underset{i}{1}\dots\underset{n}{000})$ means we are in state i .

More generally, the vector $x = (x_1, \dots, x_n)$ means the walk is in state i with relative frequency x_i .

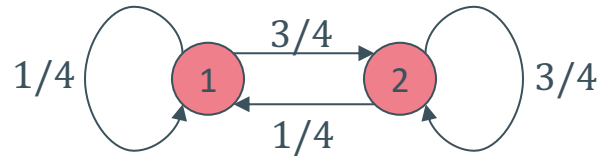
$$\sum_{i=1}^n x_i = 1$$

Change in State Vector

- If the state vector is $x = (x_1, \dots, x_n)$ 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 $x \cdot P$.

Steady State Example

- The steady state looks like a vector of probabilities $a = (a_1, \dots, a_n)$:
 - a_i is the relative frequency that we are in state i .

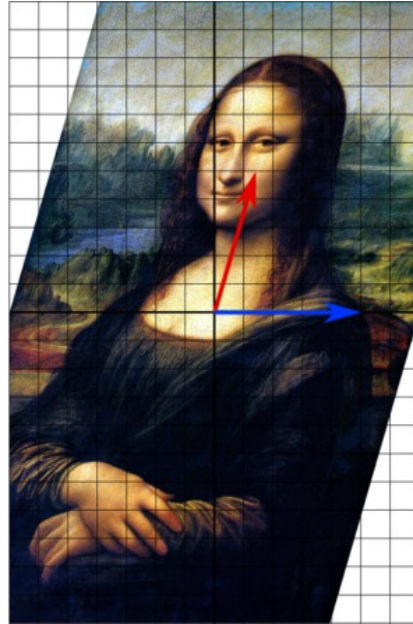
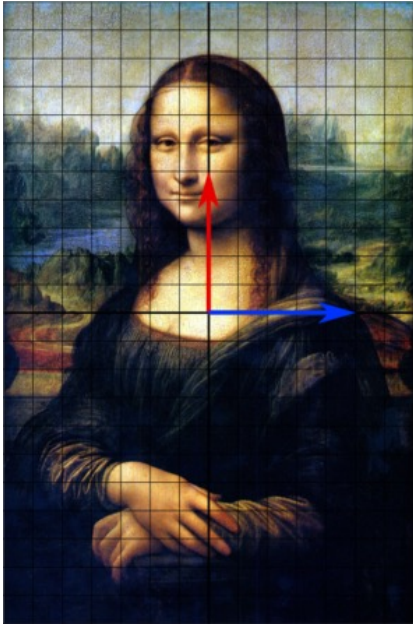


For this example, $a_1 = 1/4$ and $a_2 = 3/4$.

How do we Compute this Vector?

- Let $a = (a_1, \dots, a_n)$ denote the row vector of steady-state rates.
- If we our current position is described by a , then the next step is described as $a \cdot P$.
- But a is the steady state, so $a = a \cdot P$.
- 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 matrices always have largest eigenvalue 1.

Eigenvectors and Eigenvalues $Mx = \lambda x$



One Way of Computing a

- Recall, regardless of where we start, we eventually reach the steady state a .
- Start with any distribution (say $x = (1, 0, \dots, 0)$).
- After one step, we are at $x \cdot P$;
- after two steps at $x \cdot P^2$, then $x \cdot P^3$ and so on.
- “Eventually” means for “large” k , $x \cdot P^k = a$.
- Algorithm: multiply x by increasing powers of P until the product looks stable.

PageRank Summary

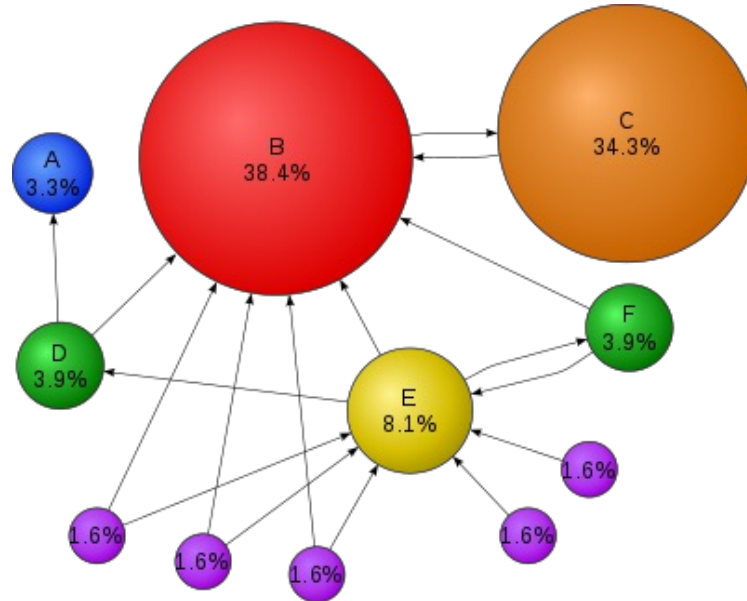
- Preprocessing:
 - Given graph of links, build matrix P
 - From it compute a
 - The entry a_i is a number between 0 and 1: the pagerank of page i .
- Query processing:
 - Retrieve pages meeting query
 - Rank them by their pagerank
 - Order is query-independent
- A variant of PageRank is used in Google, but also many other clever heuristics

PageRank: Issues and Variants

- How realistic is the random surfer model?
 - What if we modeled the back button?
 - Surfer behavior sharply skewed towards short paths
 - Search engines, bookmarks & directories make jumps non-random
- Biased Surfer Models
 - Weight edge traversal probabilities based on match with topic/query (non-uniform edge selection)
 - Bias jumps to pages on topic (e.g., based on personal bookmarks & categories of interest)

Google PageRank

- Links are also weighted according to the importance of the source node
 - Page C has a higher PageRank than Page E, even though there are fewer links to C; the one link to C comes from an important page and hence is of high value.



Community Analysis

Hyperlink-Induced Topic Search

Hyperlink-Induced Topic Search (HITS)

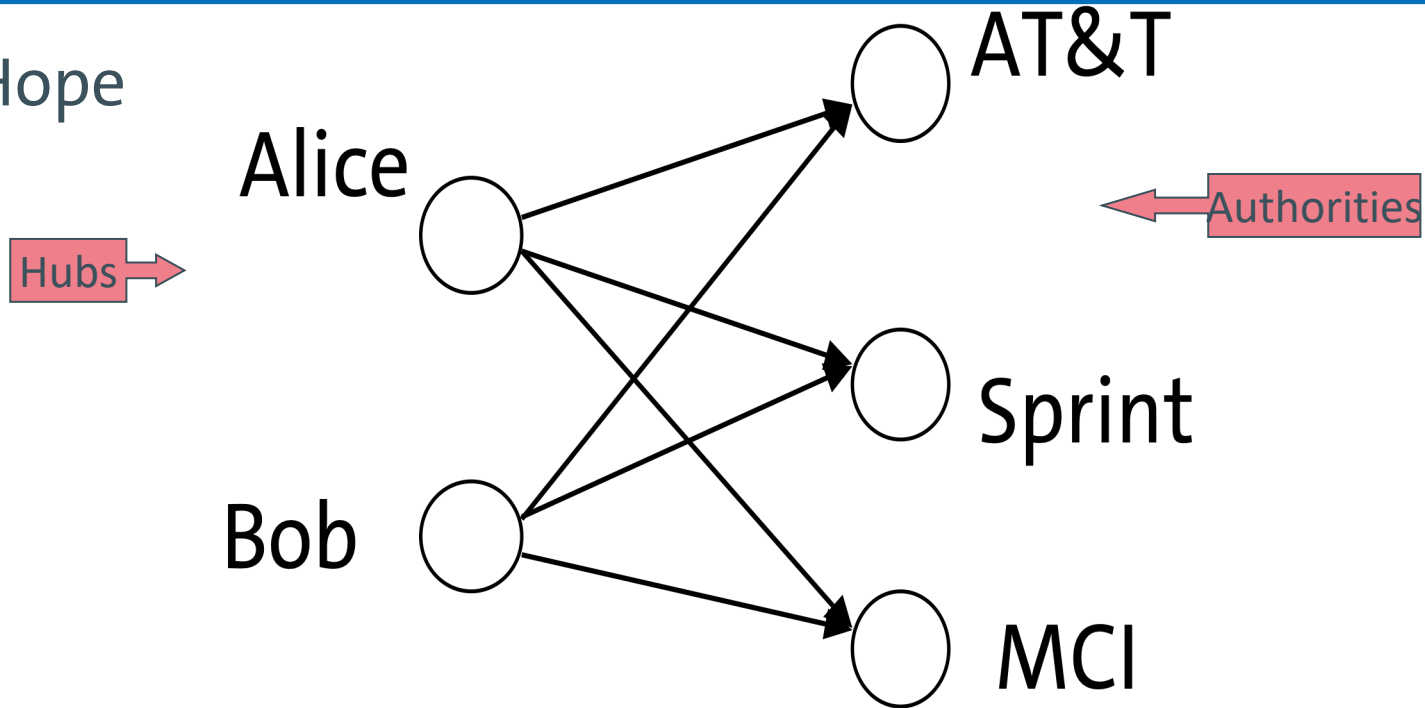
- In response to a query, instead of an ordered list of pages each matching the query, find two sets of inter-related pages:
 - **Hub pages** are good lists of links on a subject
 - e.g., “Bob’s list of cancer-related links.”
 - **Authority pages** occur recurrently on good hubs for the subject
- Best suited for “broad topic” queries rather than for page-finding queries

Jon M. Kleinberg, Hubs, Authorities, and Communities,
ACM Computing Surveys 31(4), December 1999

Hubs and Authorities

- Thus, 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

The Hope



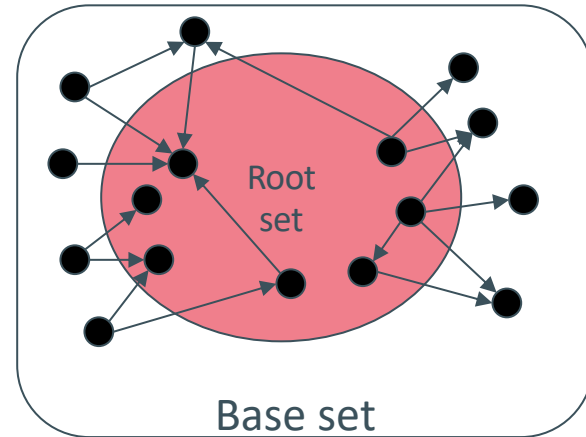
Long distance telephone companies

High-level Scheme

- Extract from the web a base set of pages that could be good hubs or authorities
- From these, identify a small set of top hub and authority pages;
→ iterative algorithm

Base Set

- Given text query (say “browser”), use a text index to get all pages containing “browser”
 - Call this the **root set** of pages
- Add in any page that either
 - points to a page in the root set, or
 - is pointed to by a page in the root set
- Call this the **base set**



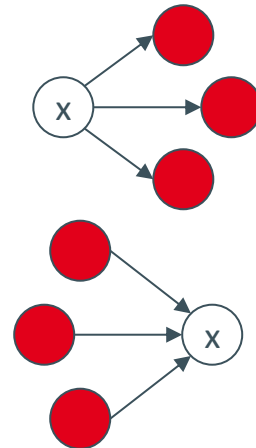
Assembling the Base Set

- Root set typically 200-1000 nodes
- Base set may have up to 5000 nodes
- How do you find the base set nodes?
 - Follow out-links by parsing root set pages
 - Get in-links (and out-links) from a connectivity server
 - Actually, suffices to text-index strings of the form href="URL" to get in-links to URL

Distilling Hubs and Authorities

Compute, for each page x in the base set, a **hub score** $h(x)$ and an **authority score** $a(x)$

- Initialize: for all x , $h(x) \stackrel{?}{=} 1$; $a(x) \leftarrow -1$;
 $h(x) \leftarrow \sum_{x \mapsto y} a(y)$; $a(x) \leftarrow \sum_{y \mapsto x} h(y)$
- Iteratively update all $h(x)$, $a(x)$;
- After iterations output pages with
 - highest $h()$ scores as top hubs
 - highest $a()$ scores as top authorities



Scaling

- To prevent the $h()$ and $a()$ values from getting too big, can scale down after each iteration
- Scaling factor does not really matter:
 - we only care about the relative values of the scores

How Many Iterations?

- Claim: relative values of scores will converge after a few iterations:
 - In fact, suitably scaled, $h()$ and $a()$ scores settle into a steady state!
- We only require the **relative orders** of the $h()$ and $a()$ scores - not their absolute values
- In practice, ~ 5 iterations get you close to stability

Deeper Understanding

- Determination of conditional dependencies between attribute values of objects in a social network
- Identification of clusters
- ...