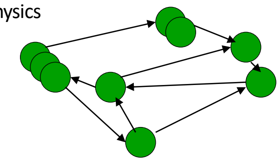
Introduction to

Information Retrieval

CS276
Information Retrieval and Web Search
Chris Manning and Pandu Nayak
Link analysis

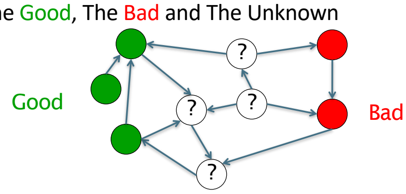
Today's lecture – hypertext and links

- We look beyond the *content* of documents
  - We begin to look at the hyperlinks between them
- Address questions like
  - Do the links represent a conferral of authority to some pages? Is this useful for ranking?
  - How likely is it that a page pointed to by the CERN home page is about high energy physics
- Big application areas
  - The Web
  - Email
  - Social networks



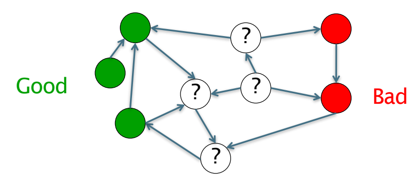
Links are everywhere

- 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



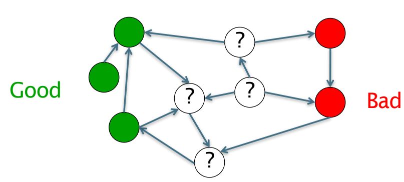
Example 1: Good/Bad/Unknown

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



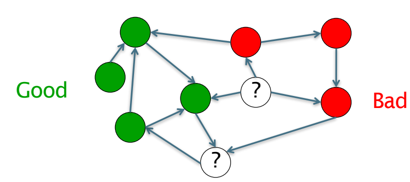
Simple iterative logic

- 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



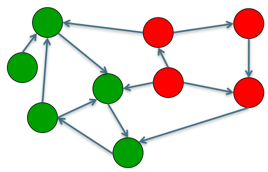
Simple iterative logic

- 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



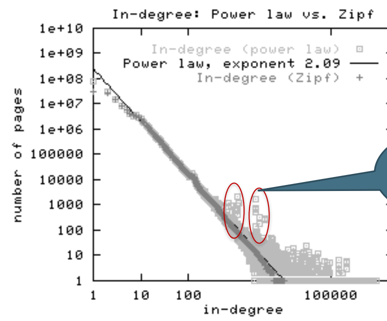
Simple iterative logic

- 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



Sometimes need probabilistic analogs – e.g., mail spam

Example 2: In-links to pages – unusual patterns ☺



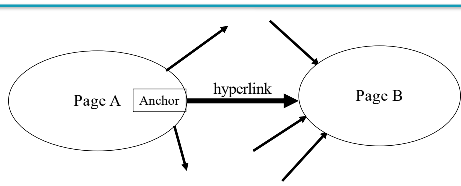
Many other examples of link analysis

- Social networks are a rich source of grouping behavior
- E.g., Shoppers' affinity – Goel+Goldstein 2010
  - Consumers whose friends spend a lot, spend a lot themselves
- http://www.cs.cornell.edu/home/kleinber/networks-book/
- See cs224w

Our primary interest in this course

- Link analysis additions to IR functionality thus far based purely on text
  - Scoring and ranking
  - Link-based clustering – topical structure from links
  - Links as features in classification – documents that link to one another are likely to be on the same subject
- Crawling
  - Based on the links seen, where do we crawl next?

The Web as a Directed Graph

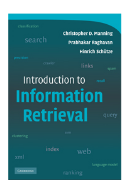


Hypothesis 1: A hyperlink between pages denotes a conferral of authority (quality signal)

Hypothesis 2: The text in the anchor of a hyperlink on page A describes the target page B

Assumption 1: reputed sites

Introduction to Information Retrieval



This is the companion website for the following book.

Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, *Introduction to Informat*

You can order this book at CUP, at your local bookstore or on the internet. The best search

The book aims to provide a modern approach to information retrieval from a computer scie
University and at the University of Stuttgart.

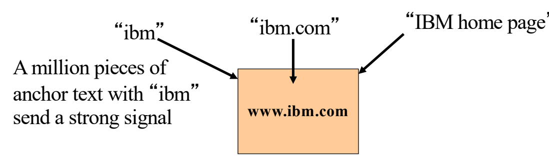
We'd be pleased to get feedback about how this book works out as a textbook, what is m
comments to: informationretrieval (at) yahoogroups (dot) com

Assumption 2: annotation of target

Anchor Text

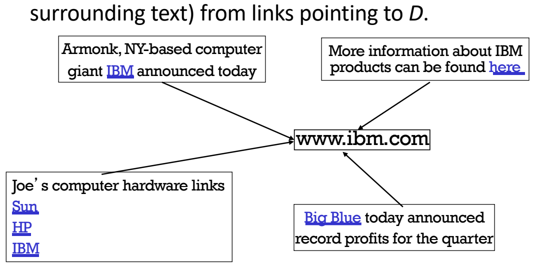
WWW Worm - McBryan [Mcbr94]

- 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.)



Indexing anchor text

- When indexing a document *D*, include (with some weight) anchor text (and perhaps nearby surrounding text) from links pointing to *D*.



Indexing anchor text

- Can sometimes have unexpected effects, e.g., spam, **miserable failure**
- Can score anchor text with weight depending on the authority of the anchor page's website
  - E.g., if we were to assume that content from cnn.com or yahoo.com is authoritative, then trust (more) the anchor text from them
  - Increase the weight of off-site anchors (non-nepotistic scoring)

Connectivity servers

Getting at all that link information inexpensively

Connectivity Server

- Support for fast queries on the web graph
  - Which URLs point to a given URL?
  - Which URLs does a given URL point to?

Stores mappings in memory from

- URL to outlinks, URL to inlinks

- Applications
  - Link analysis
  - Web graph analysis
    - Connectivity, crawl optimization
  - Crawl control

Boldi and Vigna 2004

- http://www2004.org/proceedings/docs/1p595.pdf
- Webgraph – set of algorithms and a java implementation
- Fundamental goal – maintain node adjacency lists in memory
  - For this, compressing the adjacency lists is the critical component

Adjacency lists

- The set of neighbors of a node
- Assume each URL represented by an integer
- E.g., for a 4 billion page web, need 32 bits per node … and now there are definitely > 4B pages
- Naively, this demands 64 bits to represent each hyperlink
- Boldi/Vigna get down to an average of ~3 bits/link
  - Further work achieves 2 bits/link

Adjacency list compression

- Properties exploited in compression:
  - Similarity (between lists)
  - Locality (many links from a page go to “nearby” pages)
  - Use gap encoding in sorted lists
  - Distribution of gap values

Main ideas of Boldi/Vigna

- Consider lexicographically ordered list of all URLs, e.g.,
  - www.stanford.edu/alchemy
  - www.stanford.edu/biology
  - www.stanford.edu/biology/plant
  - www.stanford.edu/biology/plant/copyright
  - www.stanford.edu/biology/plant/people
  - www.stanford.edu/chemistry

Boldi/Vigna

- Each of these URLs has an adjacency list
- Main idea: due to templates, the adjacency list of a node is similar to one of the 7 preceding URLs in the lexicographic ordering … or else encoded anew
- Express adjacency list in terms of one of these
- E.g., consider these adjacency lists
  - 1, 2, 4, 8, 16, 32, 64
  - 1, 4, 9, 16, 25, 36, 49, 64
  - 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144
  - 1, 4, 8, 16, 25, 36, 49, 64

Encode as (–2), remove 9, add 8

Gap encodings

- Given a sorted list of integers x, y, z, …, represent by x, y-x, z-y, …
- Compress each integer using a code
  - γ code - Number of bits = $1 + 2\lfloor \lg x \rfloor$
  - δ code: …
  - Information theoretic bound: $1 + \lfloor \lg x \rfloor$ bits
  - ζ code: Works well for integers from a power law [Boldi, Vigna: Data Compression Conf. 2004]

Main advantages of BV

- Depends only on locality in a canonical ordering
  - Lexicographic ordering works well for the web
- Adjacency queries can be answered very efficiently
  - To fetch out-neighbors, trace back the chain of prototypes
  - This chain is typically short in practice (since similarity is mostly intra-host)
  - Can also explicitly limit the length of the chain during encoding
- Easy to implement one-pass algorithm

Link analysis: Pagerank

Citation Analysis

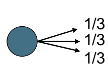
- Citation frequency
- Bibliographic coupling frequency
  - Articles that co-cite the same articles are related
- Citation indexing
  - Who is this author cited by? (Garfield 1972)
- Pagerank preview: Pinsker and Narin ’60s
  - Asked: which journals are authoritative?

The web isn’t scholarly citation

- Millions of participants, each with self interests
- Spamming is widespread
- Once search engines began to use links for ranking (roughly 1998), link spam grew
  - You can join a *link farm* – a group of websites that heavily link to one another

Pagerank scoring

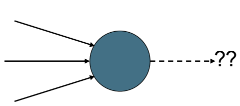
- Imagine a user doing a random walk on web pages:
  - Start at a random page
  - At each step, go out of the current page along one of the links on that page, equiprobably



- “In the long run” each page has a long-term visit rate – use this as the page’s score
- Variant: rather than equiprobable, use text and link information to have probability of following a link: intelligent surfer [Richardson and Domingos 2001]

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.
    - "Teleportation" probability
    - Simulates a web users going somewhere else
    - Solves linear algebra problems….

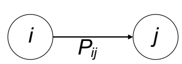
Result of teleporting

- Now cannot get stuck locally.
- There is a long-term rate at which any page is visited (not obvious, will show this).
- How do we compute this visit rate?

Markov chains

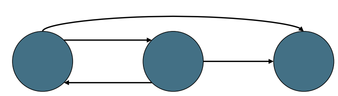
- A Markov chain consists of n states, plus an $n \times n$ transition probability matrix **P**.
- At each step, we are in one of the states.
- For $1 \le i,j \le n$, the matrix entry P_{ij} tells us the probability of j being the next state, given we are currently in state i.

$P_{ii}>0$ is OK.



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

- 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 doesn't matter where we start.

- Ergodic: no periodic patterns
  - Teleportation ensures ergodicity

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.
  $1 \quad i \quad n$

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

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

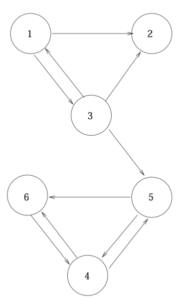
Change in probability vector

- 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 prob. 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?
  - Running this and finding out is "the power method"
    - It's actually the method of choice, done with sparse P

How do we compute this vector?

- Let $\mathbf{a} = (a_1, \ldots a_n)$ 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. (See: Perron-Frobenius theorem.)
  - Transition probability matrices always have largest eigenvalue 1.

Example: Mini web graph



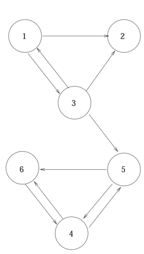
$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Example: Fixing sinks and teleporting

$$\bar{\mathbf{P}} = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{e}^T/n = \begin{pmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{pmatrix}$$

Example: Doing power iteration




```
import numpy as np
```

```
x0 = np.matrix([1/6, 1/6, 1/6, 1/6, 1/6, 1/6])
```

```
P = np.matrix([[1/60, 7/15, 7/15, 1/60, 1/60, 1/60],
               [1/6, 1/6, 1/6, 1/6, 1/6, 1/6],
               [19/60, 19/60, 1/60, 1/60, 19/60, 1/60],
               [1/60, 1/60, 1/60, 1/60, 7/15, 7/15],
               [1/60, 1/60, 1/60, 7/15, 1/60, 7/15],
               [1/60, 1/60, 1/60, 11/12, 1/60, 1/60]])
```

```
print(x0 * P)
```

```
[[0.09166667 0.16666667 0.11666667 0.26666667 0.16666667 0.19166667]]
```

```
print(x0 * P * P)
```

```
[[0.07666667 0.11791667 0.08291667 0.28916667 0.19666667 0.23666667]]
```

```
print(x0 * P * P * P * P)
```

```
[[0.05138229 0.07803542 0.05737917 0.34361667 0.20251667 0.26706979]]
```

```
print(x0 * P * P * P * P * P * P * P * P)
```

```
[[0.0391419  0.05730065 0.04374176 0.37100521 0.20527182 0.28353866]]
```

```
print(x0 * P * P * P * P * P * P * P * P * P * P * P * P * P * P * P * P)
```

```
[[0.03724891 0.05402154 0.04154868 0.37500616 0.20598094 0.28619378]]
```


Link analysis: HITS

Kleinberg (1999)

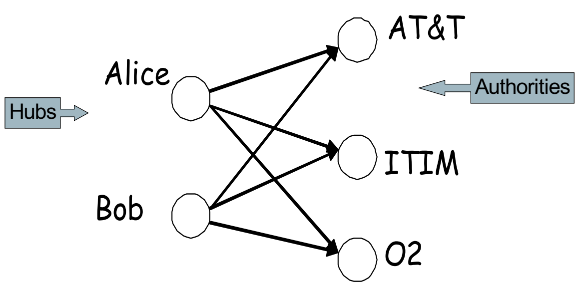
Hyperlink-Induced Topic Search (HITS)

- In response to a query, instead of an ordered list of pages each meeting the query, find <u>two</u> 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
- Gets at a broader slice of common *opinion*

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



Mobile telecom companies

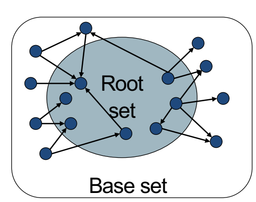
High-level scheme

- Extract from the web a <u>base set</u> 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 <u>root set</u> 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 <u>base set.</u>

Visualization



Get in-links (and out-links) from a *connectivity server*

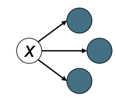
Distilling hubs and authorities

- 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)*←*1; a(x)* ←*1;*
- Iteratively update all *h(x), a(x)*; Key
- After iterations
  - output pages with highest *h()* scores as top hubs
  - highest *a()* scores as top authorities.

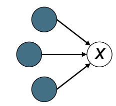
Iterative update

- Repeat the following updates, for all *x*:

$$h(x) \leftarrow \sum_{x \mapsto y} a(y)$$



$$a(x) \leftarrow \sum_{y \mapsto x} h(y)$$



Scaling

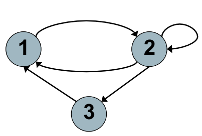
- To prevent the *h()* and *a()* values from getting too big, can scale down after each iteration.
- Scaling factor doesn't really matter:
  - we only care about the *relative* values of the scores.

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!
  - proof of this comes later.
- In practice, ~5 iterations get you close to stability.

Proof of convergence

- $n \times n$ adjacency matrix **A**:
  - each of the *n* pages in the base set has a row and column in the matrix.
  - Entry $A_{ij} = 1$ if page *i* links to page *j*, else = 0.



	1	2	3
1	0	1	0
2	1	1	1
3	1	0	0

Hub/authority vectors

- View the hub scores *h()* and the authority scores *a()* as vectors with *n* components.
- Recall the iterative updates

$$h(x) \leftarrow \sum_{x \mapsto y} a(y)$$

$$a(x) \leftarrow \sum_{y \mapsto x} h(y)$$

Rewrite in matrix form

- **h**=**Aa**.
- **a**=$\mathbf{A}^T$**h**.

Recall A^T is the transpose of A.

Substituting, **h**=$\mathbf{AA}^T\mathbf{h}$ and **a**=$\mathbf{A}^T\mathbf{Aa}$.
Thus, **h** is an eigenvector of $\mathbf{AA}^T$ and **a** is an eigenvector of $\mathbf{A}^T\mathbf{A}$.

Further, our algorithm is a particular, known algorithm for computing eigenvectors: again, the *power iteration* method.

Guaranteed to converge.

Example authorities found

- (java) Authorities
  - .328 http://www.gamelan.com/ Gamelan
  - .251 http://java.sun.com/ JavaSoft Home Page
  - .190 http://www.digitalfocus.com/... The Java Developer: How Do I ...
  - .190 http://lightyear.ncsa.uiuc.edu/;srp/java/ javabooks.html
  - .183 http://sunsite.unc.edu/javafaq/javafaq.html comp.lang.java FAQ
- (censorship) Authorities
  - .378 http://www.eff.org/ EFFweb—The Electronic Frontier Foundation
  - .344 http://www.eff.org/blueribbon.html The Blue Ribbon Campaign for Online Free Speech
  - .238 http://www.cdt.org/ The Center for Democracy and Technology
  - .235 http://www.vtw.org/ Voters Telecommunications Watch
  - .218 http://www.aclu.org/ ACLU: American Civil Liberties Union

Issues

- Topic Drift
  - Off-topic pages can cause off-topic "authorities" to be returned
    - E.g., the neighborhood graph can be about a "super topic"
- Mutually Reinforcing Affiliates
  - Affiliated pages/sites can boost each others' scores
    - Linkage between affiliated pages is not a useful signal

Resources

- IIR Chap 21
- Kleinberg, Jon (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM*. **46** (5): 604–632.
- http://www2004.org/proceedings/docs/1p309.pdf
- http://www2004.org/proceedings/docs/1p595.pdf
- http://www2003.org/cdrom/papers/refereed/p270/kamvar-270-xhtml/index.html
- http://www2003.org/cdrom/papers/refereed/p641/xhtml/p641-mccurley.html
- The WebGraph framework I: Compression techniques (Boldi et al. 2004)