Lecture 4: Information Retrieval and Web Mining

http://www.cs.kent.edu/~jin/advdatabases.html
Outline

- Information Retrieval
  - ★ Chapter 19 (Database System Concepts)
- Web Mining (Mining the Web, Soumen Chakrabarti)
- PageRank
  - ★ One of the key techniques that contributes to google’s initial success
Chapter 19: Information Retrieval

- Relevance Ranking Using Terms
- Relevance Using Hyperlinks
- Synonyms., Homonyms, and Ontologies
- Indexing of Documents
- Measuring Retrieval Effectiveness
- Information Retrieval and Structured Data
Information Retrieval Systems

- **Information retrieval (IR)** systems use a simpler data model than database systems
  
  - Information organized as a collection of documents
  - Documents are unstructured, no schema

- Information retrieval locates relevant documents, on the basis of user input such as keywords or example documents
  
  - e.g., find documents containing the words “database systems”

- Can be used even on textual descriptions provided with non-textual data such as images

- Web search engines are the most familiar example of IR systems
Information Retrieval Systems (Cont.)

- Differences from database systems
  - IR systems don’t deal with transactional updates (including concurrency control and recovery)
  - Database systems deal with structured data, with schemas that define the data organization
  - IR systems deal with some querying issues not generally addressed by database systems
    - Approximate searching by keywords
    - Ranking of retrieved answers by estimated degree of relevance
Keyword Search

In **full text** retrieval, all the words in each document are considered to be keywords.

- We use the word **term** to refer to the words in a document

Information-retrieval systems typically allow query expressions formed using keywords and the logical connectives **and, or, and not**

- **Ands** are implicit, even if not explicitly specified

Ranking of documents on the basis of estimated relevance to a query is critical

- Relevance ranking is based on factors such as
  - **Term frequency**
    - Frequency of occurrence of query keyword in document
  - **Inverse document frequency**
    - How many documents the query keyword occurs in
      - Fewer ➔ give more importance to keyword
  - **Hyperlinks to documents**
    - More links to a document ➔ document is more important
Relevance Ranking Using Terms

- **TF-IDF** (Term frequency/Inverse Document frequency) ranking:
  - Let $n(d) = \text{number of terms in the document } d$
  - $n(d, t) = \text{number of occurrences of term } t \text{ in the document } d$.
  - Relevance of a document $d$ to a term $t$
    \[
    TF(d, t) = \log \left( 1 + \frac{n(d, t)}{n(d)} \right)
    \]
  - The log factor is to avoid excessive weight to frequent terms.
  - Relevance of document to query $Q$
    \[
    r(d, Q) = \sum_{t \in Q} \frac{TF(d, t)}{n(t)}
    \]

$IDF = 1/n(t)$, $n(t)$ is the number of documents that contain the term $t$.
Relevance Ranking Using Terms (Cont.)

- Most systems add to the above model
  - ★ Words that occur in title, author list, section headings, etc. are given greater importance
  - ★ Words whose first occurrence is late in the document are given lower importance
  - ★ Very common words such as “a”, “an”, “the”, “it” etc are eliminated
    ✔ Called stop words
  - ★ Proximity: if keywords in query occur close together in the document, the document has higher importance than if they occur far apart

- Documents are returned in decreasing order of relevance score
  - ★ Usually only top few documents are returned, not all
Review

■ What’s IR system?
■ What’s the key difference between IR system and traditional relation database system?
■ What’s keyword search?
■ What’s the main factors we considered In key word search?
  ★ How to estimate/rank the relevance of a document?
  ★ What’s TF/IDF ranking?
Similarity Based Retrieval

- Similarity based retrieval - retrieve documents similar to a given document
  - Similarity may be defined on the basis of common words
    - E.g. find $k$ terms in $A$ with highest $TF(d,t)/n(t)$ and use these terms to find relevance of other documents.
- Relevance feedback: Similarity can be used to refine answer set to keyword query
  - User selects a few relevant documents from those retrieved by keyword query, and system finds other documents similar to these.
- Vector space model: define an $n$-dimensional space, where $n$ is the number of words in the document set.
  - Vector for document $d$ goes from origin to a point whose $i^{th}$ coordinate is $TF(d,t)/n(t)$
  - The cosine of the angle between the vectors of two documents is used as a measure of their similarity.
Relevance Using Hyperlinks

- Number of documents relevant to a query can be enormous if only term frequencies are taken into account
- Using term frequencies makes “spamming” easy
  - E.g. a travel agency can add many occurrences of the words “travel” to its page to make its rank very high
- Most of the time people are looking for pages from popular sites
- Idea: use popularity of Web site (e.g. how many people visit it) to rank site pages that match given keywords
- Problem: hard to find actual popularity of site
  - How?
Relevance Using Hyperlinks (Cont.)

- Solution: use number of hyperlinks to a site as a measure of the popularity or prestige of the site
  - ★ Count only one hyperlink from each site (why?)
  - ★ Popularity measure is for site, not for individual page
    - ✔ But, most hyperlinks are to root of site
    - ✔ Also, concept of “site” difficult to define since a URL prefix like cs.kent.edu contains many unrelated pages of varying popularity

- Refinements
  - ★ When computing prestige based on links to a site, give more weight to links from sites that themselves have higher prestige
    - ✔ Definition is circular
    - ✔ Set up and solve system of simultaneous linear equations
  - ★ Above idea is basis of the Google PageRank ranking mechanism
Relevance Using Hyperlinks (Cont.)

- Connections to social networking theories that ranked prestige of people
  - E.g. the president of the U.S.A has a high prestige since many people know him
  - Someone known by multiple prestigious people has high prestige

- Hub and authority based ranking
  - A **hub** is a page that stores links to many pages (on a topic)
  - An **authority** is a page that contains actual information on a topic
  - Each page gets a **hub prestige** based on prestige of authorities that it points to
  - Each page gets an **authority prestige** based on prestige of hubs that point to it
  - Again, prestige definitions are cyclic, and can be got by solving linear equations
  - Use authority prestige when ranking answers to a query
Review

- What’s IR system?
- What’s the key difference between IR system and traditional relation database system?
- What’s keyword search?
- What’s the main factors we considered in keyword search?
  - How to estimate/rank the relevance of a document?
  - What’s TF/IDF ranking?
- Methods for similarity-based search
- Relevance Using Hyperlinks
Synonyms and Homonyms

- **Synonyms**
    - ✔ need to realize that “maintenance” and “repair” are synonyms
  - ★ System can **extend query** as “motorcycle and (repair or maintenance)”

- **Homonyms**
  - E.g. “object” has different meanings as noun/verb
  - ★ Can disambiguate meanings (to some extent) from the context
  - ★ Extending queries automatically using synonyms can be problematic
    - ✔ Need to understand intended meaning in order to infer synonyms
    - ✔ Or verify synonyms with user
  - ★ Synonyms may have other meanings as well
Concept-Based Querying

- Approach
  - ★ For each word, determine the concept it represents from context
  - ★ Use one or more ontologies:
    - ✔ Hierarchical structure showing relationship between concepts
    - ✔ E.g.: the ISA relationship that we saw in the E-R model
- This approach can be used to standardize terminology in a specific field
- Ontologies can link multiple languages
- Foundation of the Semantic Web (not covered here)
Indexing of Documents

- An inverted index maps each keyword $K_i$ to a set of documents $S_i$ that contain the keyword
  - ★ Documents identified by identifiers

- Inverted index may record
  - ★ Keyword locations within document to allow proximity based ranking
  - ★ Counts of number of occurrences of keyword to compute TF

- **and** operation: Finds documents that contain all of $K_1, K_2, \ldots, K_n$.
  - ★ Intersection $S_1 \cap S_2 \cap \ldots \cap S_n$

- **or** operation: documents that contain at least one of $K_1, K_2, \ldots, K_n$
  - ★ union, $S_1 \cup S_2 \cup \ldots \cup S_n$

- Each $S_i$ is kept sorted to allow efficient intersection/union by merging
  - ★ “not” can also be efficiently implemented by merging of sorted lists
Word-Level Inverted File

<table>
<thead>
<tr>
<th>Document</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pease porridge hot, pease porridge cold,</td>
</tr>
<tr>
<td>2</td>
<td>Pease porridge in the pot,</td>
</tr>
<tr>
<td>3</td>
<td>Nine days old.</td>
</tr>
<tr>
<td>4</td>
<td>Some like it hot, some like it cold,</td>
</tr>
<tr>
<td>5</td>
<td>Some like it in the pot,</td>
</tr>
<tr>
<td>6</td>
<td>Nine days old.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number</th>
<th>Term</th>
<th>(Document; Words)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>cold</td>
<td>(1; 6), (4; 8)</td>
</tr>
<tr>
<td>2</td>
<td>days</td>
<td>(3; 2), (6; 2)</td>
</tr>
<tr>
<td>3</td>
<td>hot</td>
<td>(1; 3), (4; 4)</td>
</tr>
<tr>
<td>4</td>
<td>in</td>
<td>(2; 3), (5; 4)</td>
</tr>
<tr>
<td>5</td>
<td>it</td>
<td>(4; 3, 7), (5; 3)</td>
</tr>
<tr>
<td>6</td>
<td>like</td>
<td>(4; 2, 6), (5; 2)</td>
</tr>
<tr>
<td>7</td>
<td>nine</td>
<td>(3; 1), (6; 1)</td>
</tr>
<tr>
<td>8</td>
<td>old</td>
<td>(3; 3), (6; 3)</td>
</tr>
<tr>
<td>9</td>
<td>pease</td>
<td>(1; 1, 4), (2; 1)</td>
</tr>
<tr>
<td>10</td>
<td>porridge</td>
<td>(1; 2, 5), (2; 2)</td>
</tr>
<tr>
<td>11</td>
<td>pot</td>
<td>(2; 5), (5; 6)</td>
</tr>
<tr>
<td>12</td>
<td>some</td>
<td>(4; 1, 5), (5; 1)</td>
</tr>
<tr>
<td>13</td>
<td>the</td>
<td>(2; 4), (5; 5)</td>
</tr>
</tbody>
</table>
Measuring Retrieval Effectiveness

- Information-retrieval systems save space by using index structures that support only approximate retrieval. May result in:
  - ★ false negative (false drop) - some relevant documents may not be retrieved.
  - ★ false positive - some irrelevant documents may be retrieved.
  - ★ For many applications a good index should not permit any false drops, but may permit a few false positives.

- Relevant performance metrics:
  - ★ precision - what percentage of the retrieved documents are relevant to the query.
  - ★ recall - what percentage of the documents relevant to the query were retrieved.
Measuring Retrieval Effectiveness (Cont.)

■ Recall vs. precision tradeoff:
  ✔ Can increase recall by retrieving many documents (down to a low level of relevance ranking), but many irrelevant documents would be fetched, reducing precision

■ Measures of retrieval effectiveness:
  ★ Recall as a function of number of documents fetched, or
  ★ Precision as a function of recall
    ✔ Equivalently, as a function of number of documents fetched
    ★ E.g. “precision of 75% at recall of 50%, and 60% at a recall of 75%”

■ Problem: which documents are actually relevant, and which are not
Outline

■ Information Retrieval
  ★ Chapter 19 (Database System Concepts)

■ Web Mining
  ★ What is web mining?
  ★ Structures of WWW
  ★ Searching the Web
  ★ Web Directory
  ★ Web Mining topics

■ PageRank
  ★ One of the key techniques that help google succeed
What is Web Mining?

- Discovering useful information from the World-Wide Web and its usage patterns

- Applications
  - Web search e.g., Google, Yahoo,…
  - Vertical Search e.g., FatLens, Become,…
  - Recommendations e.g., Amazon.com
  - Advertising e.g., Google, Yahoo
  - Web site design e.g., landing page optimization
How does it differ from “classical” Data Mining?

- The web is not a relation
  - Textual information and linkage structure
- Usage data is huge and growing rapidly
  - Google’s usage logs are bigger than their web crawl
  - Data generated per day is comparable to largest conventional data warehouses
- Ability to react in real-time to usage patterns
  - No human in the loop
The World-Wide Web

- Huge
- Distributed content creation, linking (no coordination)
- Structured databases, unstructured text, semistructured
- Content includes truth, lies, obsolete information, contradictions, …
- Our modern-day Library of Alexandria

The Web
Size of the Web

Number of pages

★ Technically, infinite
  ✔ Because of dynamically generated content
  ✔ Lots of duplication (30-40%)
★ Best estimate of “unique” static HTML pages comes from search engine claims
  ✔ Google = 8 billion, Yahoo = 20 billion
  ✔ Lots of marketing hype

Number of unique web sites

★ Netcraft survey says 76 million sites (http://news.netcraft.com/archives/web_server_survey.html)
The web as a graph

- Pages = nodes, hyperlinks = edges
  - Ignore content
  - Directed graph

- High linkage
  - 8-10 links/page on average
  - Power-law degree distribution
Power-law degree distribution

Source: Broder et al, 2000
Power-laws galore

- In-degrees
- Out-degrees
- Number of pages per site
- Number of visitors
- Let’s take a closer look at structure
  - Broder et al. (2000) studied a crawl of 200M pages and other smaller crawls
  - Bow-tie structure
    - Not a “small world”
Bow-tie Structure

Source: Broder et al, 2000
Searching the Web

Content aggregators

The Web

Content consumers
Ads vs. search results

GEICO Car Insurance. Get an auto insurance quote and save today...
GEICO auto insurance, online car insurance quote, motorcycle insurance quote, online insurance sales and service from a leading insurance company.
www.geico.com/- 21k - Sep 22, 2005 - Cached - Similar pages
Auto Insurance - Buy Auto Insurance
Contact Us - Make a Payment
More results from www.geico.com >

Geico, Google Settle Trademark Dispute
The case was resolved out of court, so advertisers are still left without legal guidance on use of trademarks within ads or as keywords.

Google and GEICO settle AdWords dispute | The Register
Google and car insurance firm GEICO have settled a trade mark dispute over ... Car insurance firm GEICO sued both Google and Yahoo! subsidiary Overture in ...
www.theregister.co.uk/2005/09/09/google_geico_settlement/ - 21k - Cached - Similar pages

GEICO v. Google
... involving a lawsuit filed by Government Employees Insurance Company (GEICO). GEICO has filed suit against two major Internet search engine operators, ...
www.consumeraffairs.com/news04/geico_google.html - 19k - Cached - Similar pages
Ads vs. search results

- Search advertising is the revenue model
  - Multi-billion-dollar industry
  - Advertisers pay for clicks on their ads

- Interesting problems
  - How to pick the top 10 results for a search from 2,230,000 matching pages?
  - What ads to show for a search?
  - If I’m an advertiser, which search terms should I bid on and how much to bid?
Sidebar: What’s in a name?

- Geico sued Google, contending that it owned the trademark “Geico”
  - ★ Thus, ads for the keyword geico couldn’t be sold to others
- Court Ruling: search engines can sell keywords including trademarks
- No court ruling yet: whether the ad itself can use the trademarked word(s)
The Long Tail

Source: Chris Anderson (2004)

Sources: Erik Brynjolfsson and Jeffrey Hu, MIT, and Michael Smith, Carnegie Mellon; Barnes & Noble; Netflix; RealNetworks
The Long Tail

- Shelf space is a scarce commodity for traditional retailers
  - Also: TV networks, movie theaters,…
- The web enables near-zero-cost dissemination of information about products
- More choices necessitate better filters
  - Recommendation engines (e.g., Amazon)
  - How Into Thin Air made Touching the Void a bestseller
Web search basics

The Web

Web crawler

Indexer

Indexes

Ad indexes

Search

User

Web

Results 1-10 of about 7,310,000 for miele. (0.12 seconds)

Miele, Inc -- Anything else is a compromise
At the heart of your home, Appliances by Miele. USA.

www.miele.com/ - 20k - Cached - Similar pages

Miele Welcome to Miele, the home of the very best appliances and kitchens in the world.

www.miele.co.uk/ - 3k - Cached - Similar pages

Miele - Deutscher Hersteller von Einbaugeräten, Hausgeräten - [Translate this page]
Das Portal zum Thema Essen & Geniessen online unter www.zu-tisch.de. Miele weltweit ein Leben lang.
Wählen Sie die Miele Vertretung Ihres Landes.

www.miele.de/ - 10k - Cached - Similar pages

Herzlich willkommen bei Miele Österreich - [Translate this page]
Wenn Sie nicht automatisch weitergeleitet werden, klicken Sie bitte hier! HAUSHALTSGERÄTE

www.miele.at/ - 3k - Cached - Similar pages

Sponsored Links

CG Appliance Express
Discount Appliances (650) 756-3931
Same Day Certified Installation

www.cgappliance.com

San Francisco-Oakland-San Jose, CA

Miele Vacuum Cleaners
Miele Vacuums- Complete Selection
Free Shipping!

www.best-vacuum.com

36
Search engine components

- **Spider** (a.k.a. crawler/robot) – builds corpus
  - ★ Collects web pages recursively
    - ✔ For each known URL, fetch the page, parse it, and extract new URLs
    - ✔ Repeat
  - ★ Additional pages from direct submissions & other sources

- **The indexer** – creates inverted indexes
  - ★ Various policies wrt which words are indexed, capitalization, support for Unicode, stemming, support for phrases, etc.

- **Query processor** – serves query results
  - ★ Front end – query reformulation, word stemming, capitalization, optimization of Booleans, etc.
  - ★ Back end – finds matching documents and ranks them
Web Search Engines

- **Web crawlers** are programs that locate and gather information on the Web
  - Recursively follow hyperlinks present in known documents, to find other documents
    - ✔ Starting from a *seed* set of documents
  - ✔ Fetched documents
    - ✔ Handed over to an indexing system
    - ✔ Can be discarded after indexing, or store as a *cached* copy
- Crawling the entire Web would take a very large amount of time
  - ✔ Search engines typically cover only a part of the Web, not all of it
  - ✔ Take months to perform a single crawl
Web Crawling (Cont.)

- Crawling is done by multiple processes on multiple machines, running in parallel
  - Set of links to be crawled stored in a database
  - New links found in crawled pages added to this set, to be crawled later

- Indexing process also runs on multiple machines
  - Creates a new copy of index instead of modifying old index
  - Old index is used to answer queries
  - After a crawl is “completed” new index becomes “old” index

- Multiple machines used to answer queries
  - Indices may be kept in memory
  - Queries may be routed to different machines for load balancing
Directories

- Storing related documents together in a library facilitates browsing ★ users can see not only requested document but also related ones.
- Browsing is facilitated by classification system that organizes logically related documents together.
- Organization is hierarchical: **classification hierarchy**
A Classification Hierarchy For A Library System

- books
  - science
    - math
    - computer science
  - engineering
  - fiction
  - algorithms
    - graph algorithms
Classification DAG

- Documents can reside in multiple places in a hierarchy in an information retrieval system, since physical location is not important.
- Classification hierarchy is thus Directed Acyclic Graph (DAG)
A Classification DAG For A Library Information Retrieval System

- books
  - science
  - engineering
  - fiction
  - math
    - computer science
  - algorithms
    - graph algorithms
A **Web directory** is just a classification directory on Web pages

- E.g. Yahoo! Directory, Open Directory project
- **Issues:**
  - What should the directory hierarchy be?
  - Given a document, which nodes of the directory are categories relevant to the document
- Often done manually
  - Classification of documents into a hierarchy may be done based on term similarity
Web Mining topics

- Crawling the web
- Web graph analysis
- Structured data extraction
- Classification and vertical search
- Collaborative filtering
- Web advertising and optimization
- Mining web logs
- Systems Issues
Extracting structured data

http://www.fatlens.com
Extracting Structured Data

Software Implementation Consultant / Engineer
Kaidara Software (Los Altos, CA)
Kaidara Software (www.kaidara.com) provides software solutions that enable firms to effectively harness the experience and know-how within an organization to reduce the cost of delivering superior customer service. We are looking for a Software Implementation Consultant / Engineer to add to our...
2 days and 3 hours ago from Monster

Software Engineer
ESP Enviromental Software (Mountain View, CA)
... server-side data updates and various data manipulation tools. You'll participate in the design and development of Internet/Intranet application software to deliver the next generation of our products line that allows our customers to engage in business-to-business, e-commerce and global...
2 days and 19 hours ago from Dice

http://www.simplyhired.com
Information Retrieval and Structured Data

- Information retrieval systems originally treated documents as a collection of words

- Information extraction systems infer structure from documents, e.g.:
  - Extraction of house attributes (size, address, number of bedrooms, etc.) from a text advertisement
  - Extraction of topic and people named from a new article

- Relations or XML structures used to store extracted data
  - System seeks connections among data to answer queries
  - Question answering systems
**Intuition**: solve the recursive equation: “a page is important if important pages link to it.”

In high-falutin’ terms: importance = the principal eigenvector of the stochastic matrix of the Web.

★ A few fixups needed.
Enumerate pages.

Page $i$ corresponds to row and column $i$.

$M[i,j] = 1/n$ if page $j$ links to $n$ pages, including page $i$; 0 if $j$ does not link to $i$.

$M[i,j]$ is the probability we’ll next be at page $i$ if we are now at page $j$. 

Stochastic Matrix of the Web
Suppose page $j$ links to 3 pages, including $i$. 

![Diagram showing page $i$ linked by page $j$ with weight $1/3$.]
Suppose $\mathbf{v}$ is a vector whose $i^{th}$ component is the probability that we are at page $i$ at a certain time.

If we follow a link from $i$ at random, the probability distribution for the page we are then at is given by the vector $M\mathbf{v}$. 
Starting from any vector $v$, the limit $M(M(\ldots M(Mv)\ldots))$ is the distribution of page visits during a random walk.

Intuition: pages are important in proportion to how often a random walker would visit them.

The math: limiting distribution = principal eigenvector of $M = \text{PageRank}$. 
Example: The Web in 1839

Yahoo

Amazon

M’soft

<table>
<thead>
<tr>
<th></th>
<th>y</th>
<th>a</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1/2</td>
<td>1/2</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>1/2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>m</td>
<td>0</td>
<td>1/2</td>
<td>0</td>
</tr>
</tbody>
</table>
Simulating a Random Walk

- Start with the vector $v = [1,1,\ldots,1]$ representing the idea that each Web page is given one unit of *importance*.
- Repeatedly apply the matrix $M$ to $v$, allowing the importance to flow like a random walk.
- Limit exists, but about 50 iterations is sufficient to estimate final distribution.
Example

Equations $\mathbf{v} = M \mathbf{v}$:

- $y = y/2 + a/2$
- $a = y/2 + m$
- $m = a/2$

\[
\begin{array}{cccccc}
y & 1 & 1 & 5/4 & 9/8 & 6/5 \\
a & = & 1 & 3/2 & 1 & 11/8 \ldots & 6/5 \\
m & 1 & 1/2 & 3/4 & 1/2 & 3/5 \\
\end{array}
\]
Because there are no constant terms, these 3 equations in 3 unknowns do not have a unique solution.

Add in the fact that $y + a + m = 3$ to solve.

In Web-sized examples, we cannot solve by Gaussian elimination; we need to use relaxation (= iterative solution).
Real-World Problems

- Some pages are “dead ends” (have no links out).
  ★ Such a page causes importance to leak out.

- Other (groups of) pages are *spider traps* (all out-links are within the group).
  ★ Eventually spider traps absorb all importance.
Microsoft Becomes Dead End

Yahoo

Amazon

M’soft

\[
\begin{bmatrix}
y & a & m \\
y & 1/2 & 1/2 & 0 \\
a & 1/2 & 0 & 0 \\
m & 0 & 1/2 & 0 \\
\end{bmatrix}
\]
Example

Equations $\mathbf{v} = M \mathbf{v}$:

- $y = y/2 + a/2$
- $a = y/2$
- $m = a/2$

\[
\begin{array}{ccccccc}
  y & 1 & 1 & 3/4 & 5/8 & \ldots & 0 \\
  a & = & 1 & 1/2 & 1/2 & 3/8 & \ldots & 0 \\
  m & & 1 & 1/2 & 1/4 & 1/4 & \ldots & 0 \\
\end{array}
\]
M’soft Becomes Spider Trap

Yahoo

Amazon

M’soft

\[
\begin{array}{cccc}
 y & a & m \\
 y & 1/2 & 1/2 & 0 \\
 a & 1/2 & 0 & 0 \\
 m & 0 & 1/2 & 1 \\
\end{array}
\]
Example

- Equations $v = Mv$:
  
  $y = y/2 + a/2$
  
  $a = y/2$
  
  $m = a/2 + m$

  
<table>
<thead>
<tr>
<th>y</th>
<th>1</th>
<th>1</th>
<th>3/4</th>
<th>5/8</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>1/2</td>
<td>1/2</td>
<td>3/8</td>
<td>. . .</td>
</tr>
<tr>
<td>m</td>
<td>1</td>
<td>3/2</td>
<td>7/4</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
Google Solution to Traps, Etc.

- “Tax” each page a fixed percentage at each iteration.
- Add the same constant to all pages.
- Models a random walk with a fixed probability of going to a random place next.
Example: Previous with 20% Tax

Equations: \[ v = 0.8 \left( \frac{y}{v} \right) + 0.2 \]
\[ y = 0.8 \left( \frac{y}{2} + \frac{a}{2} \right) + 0.2 \]
\[ a = 0.8 \left( \frac{y}{2} \right) + 0.2 \]
\[ m = 0.8 \left( \frac{a}{2} + m \right) + 0.2 \]

\[
\begin{array}{cccccc}
y & 1 & 1.00 & 0.84 & 0.776 & 7/11 \\
a & = & 1 & 0.60 & 0.60 & 0.536 & \ldots & 5/11 \\
m & 1 & 1.40 & 1.56 & 1.688 & 21/11 \\
\end{array}
\]
In this example, because there are no dead-ends, the total importance remains at 3.

In examples with dead-ends, some importance leaks out, but total remains finite.
Because there are constant terms, we can expect to solve small examples by Gaussian elimination.

Web-sized examples still need to be solved by relaxation.
Newton-like prediction of where components of the principal eigenvector are heading.

Take advantage of locality in the Web.

Each technique can reduce the number of iterations by 50%.

- Important --- PageRank takes time!
Three consecutive values for the importance of a page suggests where the limit might be.

Guess for the next round
Exploiting Substructure

- Pages from particular domains, hosts, or paths, like stanford.edu or www-db.stanford.edu/~ullman tend to have higher density of links.

- Initialize PageRank using ranks within your local cluster, then ranking the clusters themselves.
Strategy

- Compute local PageRanks (in parallel?).
- Use local weights to establish intercluster weights on edges.
- Compute PageRank on graph of clusters.
- Initial rank of a page is the product of its local rank and the rank of its cluster.
- “Clusters” are appropriately sized regions with common domain or lower-level detail.
In Pictures

Local ranks

Intercluster weights

Ranks of clusters

Initial eigenvector

1.5

2.05

3.0

2.0

0.15

0.1

0.05
Hubs and Authorities

- Mutually recursive definition:
  - ★ A *hub* links to many authorities;
  - ★ An *authority* is linked to by many hubs.

- Authorities turn out to be places where information can be found.
  - ★ Example: course home pages.

- Hubs tell where the authorities are.
  - ★ Example: CSD course-listing page.
H&A uses a matrix $A[i, j] = 1$ if page $i$ links to page $j$, 0 if not.

$A^T$, the transpose of $A$, is similar to the PageRank matrix $M$, but $A^T$ has 1’s where $M$ has fractions.
### Example

**Diagram:**
- **Yahoo**
- **Amazon**
- **M’soft**

**Adjacency Matrix (A):**

<table>
<thead>
<tr>
<th></th>
<th>y</th>
<th>a</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>m</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Using Matrix $A$ for H&A

- Powers of $A$ and $A^T$ diverge in size of elements, so we need scale factors.
- Let $h$ and $a$ be vectors measuring the “hubbiness” and authority of each page.
- Equations: $h = \lambda Aa$; $a = \mu A^T h$.
  - ★ Hubbiness = scaled sum of authorities of successor pages (out-links).
  - ★ Authority = scaled sum of hubbiness of predecessor pages (in-links).
Consequences of Basic Equations

From $h = \lambda A a$; $a = \mu A^T h$ we can derive:

★ $h = \lambda \mu A A^T h$
★ $a = \lambda \mu A^T A a$

Compute $h$ and $a$ by iteration, assuming initially each page has one unit of hubbiness and one unit of authority.
★ Pick an appropriate value of $\lambda \mu$. 
Example

\[
A = \begin{bmatrix}
1 & 1 & 1 \\
1 & 0 & 1 \\
0 & 1 & 0
\end{bmatrix}
\quad A^T = \begin{bmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 1 & 0
\end{bmatrix}
\quad AA^T = \begin{bmatrix}
3 & 2 & 1 \\
2 & 2 & 0 \\
1 & 1 & 1
\end{bmatrix}
\quad A^TA = \begin{bmatrix}
2 & 1 & 2 \\
1 & 2 & 1 \\
2 & 1 & 2
\end{bmatrix}
\]

\[
a(\text{yahoo}) = 1524114\ldots 1+\sqrt{3}
\]
\[
a(\text{amazon}) = 141884\ldots 2
\]
\[
a(\text{m’soft}) = 1524114\ldots 1+\sqrt{3}
\]
\[
h(\text{yahoo}) = 1628132\ldots 1.000
\]
\[
h(\text{amazon}) = 142096\ldots 0.735
\]
\[
h(\text{m’soft}) = 12836\ldots 0.268
\]
Solving the Equations

- Solution of even small examples is tricky, because the value of $\lambda \mu$ is one of the unknowns.
  - Each equation like $y = \lambda \mu (3y + 2a + m)$ lets us solve for $\lambda \mu$ in terms of $y, a, m$; equate each expression for $\lambda \mu$.
- As for PageRank, we need to solve big examples by relaxation.
Details for \( h \) --- (1)

\[
y = \lambda \mu (3y + 2a + m)
\]
\[
a = \lambda \mu (2y + 2a)
\]
\[
m = \lambda \mu (y + m)
\]

- Solve for \( \lambda \mu \):

\[
\lambda \mu = \frac{y}{(3y + 2a + m)} = \frac{a}{(2y + 2a)} = \frac{m}{(y + m)}
\]
Details for $h$ --- (2)

- Assume $y = 1$.

$$\lambda \mu = \frac{1}{3 + 2a + m} = \frac{a}{2 + 2a} = \frac{m}{1 + m}$$

- Cross-multiply second and third:

$$a + am = 2m + 2am \text{ or } a = 2m/(1-m)$$

- Cross multiply first and third:

$$1 + m = 3m + 2am + m^2 \text{ or } a = (1 - 2m - m^2)/2m$$
Details for h --- (3)

- Equate formulas for $a$:
  \[ a = \frac{2m}{1-m} = \frac{(1-2m-m^2)}{2m} \]

- Cross-multiply:
  \[ 1 - 2m - m^2 - m + 2m^2 + m^3 = 4m^2 \]

- Solve for $m$: \( m = .268 \)

- Solve for $a$: \( a = \frac{2m}{1-m} = .735 \)
Solving H&A in Practice

- Iterate as for PageRank; don’t try to solve equations.
- But keep components within bounds.
  - ★ Example: scale to keep the largest component of the vector at 1.
- Trick: start with \( h = [1,1,\ldots,1] \); multiply by \( A^T \) to get first \( a \); scale, then multiply by \( A \) to get next \( h, \ldots \)
H&A Versus PageRank

- If you talk to someone from IBM, they will tell you “IBM invented PageRank.”
  - What they mean is that H&A was invented by Jon Kleinberg when he was at IBM.
- But these are not the same.
- H&A has been used, e.g., to analyze important research papers; it does not appear to be a substitute for PageRank.