Web Search and Beyond

Carl Meyer

Department of Mathematics
North Carolina State University
Raleigh, NC

Institute Of Advanced Analytics
Raleigh, NC
April 14, 2010
**Short History of IR**

**IR** = search within doc. coll. for particular info. need (query)

- **B. C.** cave paintings
- **12th cent. A.D.** invention of paper, monks in scriptoriums
- **1450** Gutenberg’s printing press
- **1700s** Franklin’s public libraries
- **1872** Dewey’s decimal system
- **1940s-1950s** Card catalog, Computer
System for the Mechanical Analysis and Retrieval of Text

Harvard 1962 – 1965
Cornell 1965 – 1970

- Implemented on IBM 7094 & IBM 360
- Based on matrix methods
Term–Document Matrices

Start with dictionary of terms

Words or phrases (e.g., landing gear)
Term–Document Matrices

Start with dictionary of terms

Words or phrases (e.g., landing gear)

Index Each Document

Humans scour pages and mark key terms
Term–Document Matrices

**Start with dictionary of terms**

Words or phrases (e.g., *landing gear*)

**Index Each Document**

Humans scour pages and mark key terms

Count $f_{ij} = \#$ times term $i$ appears in document $j$
Term–Document Matrices

Start with dictionary of terms

Words or phrases (e.g., landing gear)

Index Each Document

Humans scour pages and mark key terms

Count $f_{ij} =$ # times term $i$ appears in document $j$

Term–Document Matrix

\[
\begin{pmatrix}
    f_{11} & f_{12} & \cdots & f_{1n} \\
    f_{21} & f_{22} & \cdots & f_{2n} \\
    \vdots & \vdots & \ddots & \vdots \\
    f_{m1} & f_{m2} & \cdots & f_{mn}
\end{pmatrix} = A_{m \times n}
\]
Query Matching

Query Vector

\[ q^T = (q_1, q_2, \ldots, q_m) \]

\[ q_i = \begin{cases} 
1 & \text{if Term } i \text{ is requested} \\ 
0 & \text{if not} 
\end{cases} \]
Query Matching

Query Vector

\[ q^T = (q_1, q_2, \ldots, q_m) \]

\[ q_i = \begin{cases} 
1 & \text{if Term } i \text{ is requested} \\
0 & \text{if not} 
\end{cases} \]

How Close is Query to Each Document?
Query Matching

Query Vector

\[ q^T = (q_1, q_2, \ldots, q_m) \]

\[ q_i = \begin{cases} 
1 & \text{if Term } i \text{ is requested} \\
0 & \text{if not} 
\end{cases} \]

How Close is Query to Each Document?

i.e., how close is \( q \) to each column \( A_i \)?
Query Matching

Query Vector

\[ q^T = (q_1, q_2, \ldots, q_m) \]

\[ q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases} \]

How Close is Query to Each Document?

i.e., how close is \( q \) to each column \( A_i \)?

Use \( \delta_i = \cos \theta_i = \frac{q^T A_i}{\|q\| \|A_i\|} \)
Query Matching

Query Vector

\[ q^T = (q_1, q_2, \ldots, q_m) \]

\[ q_i = \begin{cases} 
1 & \text{if Term } i \text{ is requested} \\
0 & \text{if not} 
\end{cases} \]

How Close is Query to Each Document?

i.e., how close is \( q \) to each column \( A_i \)?

Use

\[ \delta_i = \cos \theta_i = \frac{q^T A_i}{\|q\| \|A_i\|} \]

Rank documents by size of \( \delta_i \)

Return Document \( i \) to user when \( \delta_i \geq tol \)
Susan Dumais’s Improvement

- Approximate $A$ with a lower rank matrix
- Effect is to compress data in $A$

- 2 patents for Bell/Telcordia

- LATENT SEMANTIC INDEXING
Latent Semantic Indexing

Use a finite Fourier expansion of $A$

$$A = \sum_{i=1}^{r} \sigma_i Z_i,$$

$$\langle Z_i | Z_j \rangle = \begin{cases} 1 & i = j, \\ 0 & i \neq j, \end{cases} \quad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = |\langle Z_i | A \rangle| = \text{amount of } A \text{ in direction of } Z_i$$
Latent Semantic Indexing

Use a finite Fourier expansion of $A$

$$A = \sum_{i=1}^{r} \sigma_i Z_i,$$

$$\langle Z_i | Z_j \rangle = \begin{cases} 1 & i = j, \\ 0 & i \neq j, \end{cases}$$

$$|\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = |\langle Z_i | A \rangle| = \text{amount of } A \text{ in direction of } Z_i$$

Realign data along dominant directions $\{Z_1, \ldots, Z_k, Z_{k+1}, \ldots, Z_r\}$

— Project $A$ onto $span \{Z_1, Z_2, \cdots, Z_k\}$
Latent Semantic Indexing

Use a finite Fourier expansion of $A$

$$A = \sum_{i=1}^{r} \sigma_i Z_i,$$

$$\langle Z_i | Z_j \rangle = \begin{cases} 1 & i=j, \\ 0 & i \neq j, \end{cases}$$

$$|\sigma_i| = |\langle Z_i | A \rangle| = \text{amount of } A \text{ in direction of } Z_i$$

Realign data along dominant directions $\{Z_1, \ldots, Z_k, Z_{k+1}, \ldots, Z_r\}$

— Project $A$ onto $\text{span} \{Z_1, Z_2, \ldots, Z_k\}$

**Truncate:** $A_k = P(A) = \sigma_1 Z_1 + \sigma_2 Z_2 + \cdots + \sigma_k Z_k$
Latent Semantic Indexing

Use a finite Fourier expansion of $A$

$$A = \sum_{i=1}^{r} \sigma_i Z_i,$$

$$\langle Z_i \vert Z_j \rangle = \begin{cases} 
1 & i=j, \\
0 & i \neq j,
\end{cases} \quad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = |\langle Z_i \vert A \rangle| = \text{amount of } A \text{ in direction of } Z_i$$

Realign data along dominant directions $\{Z_1, \ldots, Z_k, Z_{k+1}, \ldots, Z_r\}$

— Project $A$ onto $span \{Z_1, Z_2, \cdots, Z_k\}$

Truncate: $A_k = P(A) = \sigma_1 Z_1 + \sigma_2 Z_2 + \cdots + \sigma_k Z_k$

LSI: Query matching with $A_k$ in place of $A$

— $Doc_2$ forced closer to $Doc_1 \implies$ better chance of finding $Doc_2$
Latent Semantic Indexing

Use a finite Fourier expansion of $A$

$$A = \sum_{i=1}^{r} \sigma_i Z_i, \quad \langle Z_i | Z_j \rangle = \begin{cases} 1 & i=j, \\ 0 & i \neq j, \end{cases} \quad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = |\langle Z_i | A \rangle| = \text{amount of } A \text{ in direction of } Z_i$$

Realign data along dominant directions $\{Z_1, \ldots, Z_k, Z_{k+1}, \ldots, Z_r\}$

— Project $A$ onto $\text{span} \{Z_1, Z_2, \cdots, Z_k\}$

Truncate: $A_k = P(A) = \sigma_1 Z_1 + \sigma_2 Z_2 + \cdots + \sigma_k Z_k$

LSI: Query matching with $A_k$ in place of $A$

— $Doc_2$ forced closer to $Doc_1 \implies$ better chance of finding $Doc_2$

“Best” mathematical solution

— SVD: $A = UDV^T = \sum \sigma_i u_i v_i^T$

$$Z_i = u_i v_i^T$$
Strengths & Weaknesses

Pros
- Finds hidden connections
Strengths & Weaknesses

Pros

• Finds hidden connections

• Can be adapted to identify document clusters
  — Data mining applications
Strengths & Weaknesses

Pros

- Finds hidden connections
- Can be adapted to identify document clusters
  - Data mining applications
- Performs well on document collections that are
  - Small + Homogeneous + Static
Strengths & Weaknesses

Pros

- Finds hidden connections
- Can be adapted to identify document clusters
  - Data mining applications
- Performs well on document collections that are
  - Small + Homogeneous + Static

Cons

- Rankings are query dependent
  - Rank of each doc is recomputed for each query
Strengths & Weaknesses

Pros
- Finds hidden connections
- Can be adapted to identify document clusters
  - Data mining applications
- Performs well on document collections that are
  - Small + Homogeneous + Static

Cons
- Rankings are query dependent
  - Rank of each doc is recomputed for each query
- Only semantic content used
  - Susceptible to malicious manipulation
Strengths & Weaknesses

Pros

• Finds hidden connections

• Can be adapted to identify document clusters
  — Data mining applications

• Performs well on document collections that are
  ▶ Small + Homogeneous + Static

Cons

• Rankings are query dependent
  — Rank of each doc is recomputed for each query

• Only semantic content used
  — Susceptible to malicious manipulation

• Difficult to add & delete documents
Strengths & Weaknesses

Pros

• Finds hidden connections
• Can be adapted to identify document clusters
  — Data mining applications
• Performs well on document collections that are
  ▶ Small + Homogeneous + Static

Cons

• Rankings are query dependent
  — Rank of each doc is recomputed for each query
• Only semantic content used
  — Susceptible to malicious manipulation
• Difficult to add & delete documents
• Finding optimal compression requires empirical tuning
Web Documents

Different from other document collections

- It’s huge
  - Billions of pages, where average page size $\geq 500$KB
  - Many-many times the size of Library of Congress print collection
Web Documents

Different from other document collections

• It’s huge
  – Billions of pages, where average page size $\geq 500$KB
  – Many-many times the size of Library of Congress print collection

• It’s dynamic
  – 40% of all pages change in a week
  – 23% of .com pages change daily
  – Billions of pages added each year
Web Documents

Different from other document collections

• It’s huge
  – Billions of pages, where average page size $\geq$ 500KB
  – Many-many times the size of Library of Congress print collection

• It’s dynamic
  – 40% of all pages change in a week
  – 23% of .com pages change daily
  – Billions of pages added each year

• It’s self-organized
  – No standards, review process, formats
  – Errors, falsehoods, link rot, and spammers!
Web Documents

Different from other document collections

- It’s huge
  - Billions of pages, where average page size $\geq 500$KB
  - Many-many times the size of Library of Congress print collection

- It’s dynamic
  - 40% of all pages change in a week
  - 23% of .com pages change daily
  - Billions of pages added each year

- It’s self-organized
  - No standards, review process, formats
  - Errors, falsehoods, link rot, and spammers!

- It has many users
  - Google alone processes more than 620 million queries per day
Web Search Components

Web Crawlers

Software robots gather web pages
Web Search Components

Web Crawlers
Software robots gather web pages

Doc Server
Stores docs and snippits
Web Search Components

Web Crawlers

Software robots gather web pages

Doc Server

Stores docs and snippits

Index Server

Scans pages and does term indexing

Terms $\rightarrow$ Pages (similar to book index)
The Ranking Module

- Measure the importance of each page
The Ranking Module

- Measure the importance of each page
- The measure should be Independent of any query
The Ranking Module

- Measure the importance of each page
- The measure should be Independent of any query
- Compute these measures off-line before queries are processed
The Ranking Module

- Measure the importance of each page
- The measure should be Independent of any query
- Compute these measures off-line before queries are processed
- Google’s PageRank\(^\text{©}\) distinguishes it from all competitors
  - Primarily determined by the link structure of the Web
  - Tempered by some content considerations
The Ranking Module

- Measure the importance of each page
- The measure should be Independent of any query
- Compute these measures off-line before queries are processed
- Google’s PageRank\(^{\text{©}}\) distinguishes it from all competitors
  - Primarily determined by the link structure of the Web
  - Tempered by some content considerations

Google’s PageRank = Google’s $$$$$$
The Process

query ➔ Web Server
The Process

query

Web Server

Index Server
The Process
The Process

- User
- Query
- Web Server
- Index Server
- Doc Server
- Page Ranks
<table>
<thead>
<tr>
<th>Daily page views for Google.com</th>
<th>7.2 billion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monthly worldwide searches on Google sites</td>
<td>87.8 billion</td>
</tr>
<tr>
<td>Global search market share</td>
<td>85.78%</td>
</tr>
<tr>
<td>Daily visitors to Google.com</td>
<td>620 million</td>
</tr>
<tr>
<td>Google.com’s global website ranking</td>
<td>1</td>
</tr>
<tr>
<td>The amount of data processed daily by Google</td>
<td>20 PB</td>
</tr>
</tbody>
</table>

Google Search support for fictional languages:
Leetspeak (H4x0r), Klingon, Pig Latin, Elmer Fudd and Bork, bork, bork!
Business intelligence - Wikipedia, the free encyclopedia
Business intelligence (BI) is a business management term which refers to applications and technologies which are used to gather, provide access to, ... en.wikipedia.org/wiki/Business_intelligence - 43k - Cached - Similar pages

Business Intelligence.com :: The Resource for Business Intelligence
The Business Intelligence resource for business and technical professionals covering a wide range of topics including Performance Management, Data Warehouse ... www.businessintelligence.com/ - 74k - Apr 15, 2007 - Cached - Similar pages

Business Intelligence and Performance Management Software ...
Business intelligence and business performance management software. Reporting, analytics software, budgeting software, balanced scorecard software, ... 
Stock quote for COGN
www.cognos.com/ - 32k - Cached - Similar pages

Oracle Business Intelligence Solutions
The First Comprehensive, Cost-Effective BI Solution Only Oracle delivers a complete, pre-integrated technology foundation to reduce the cost and complexity ... www.oracle.com/solutions/business_intelligence/index.html - 55k - Cached - Similar pages

Business Intelligence - Management Best Practice Reports
Business Intelligence: Providers of independent reports containing best practice advice, proprietary research findings and case studies for senior managers ... www.business-intelligence.co.uk/ - 18k - Cached - Similar pages

Intelligent Enterprise: Better Insight for Business Decisions
Take Your Pick

Amount of Internet search results that Web surfers typically scan before selecting one

- First page of search results: 39%
- First two pages: 19%
- First three pages: 9%
- More than first three pages: 10%
- A few search results*: 23%

*Top results without reading through the whole page

Note: Sample size is 2,369 people
Sources: JupiterResearch; iProspect
Business intelligence - Wikipedia, the free encyclopedia
Business intelligence (BI) is a business management term which refers to applications and technologies which are used to gather, provide access to, ...
en.wikipedia.org/wiki/Business_intelligence - 43k - Cached - Similar pages

Business Intelligence.com :: The Resource for Business Intelligence
The Business Intelligence resource for business and technical professionals covering a wide range of topics including Performance Management, Data Warehouse ...
www.businessintelligence.com/ - 74k - Apr 15, 2007 - Cached - Similar pages

Business Intelligence and Performance Management Software ...
Business intelligence and business performance management software. Reporting, analytics software, budgeting software, balanced scorecard software, ...
Stock quote for COGN
www.cognos.com/ - 32k - Cached - Similar pages

Oracle Business Intelligence Solutions
The First Comprehensive, Cost-Effective BI Solution Only Oracle delivers a complete, pre-integrated technology foundation to reduce the cost and complexity ...
www.oracle.com/solutions/business_intelligence/index.html - 55k - Cached - Similar pages

Business Intelligence - Management Best Practice Reports
Business Intelligence: Providers of independent reports containing best practice advice, proprietary research findings and case studies for senior managers ...
www.business-intelligence.co.uk/ - 18k - Cached - Similar pages

Intelligent Enterprise: Better Insight for Business Decisions
**Business intelligence - Wikipedia, the free encyclopedia**

Business intelligence (BI) is a business management term which refers to applications and technologies which are used to gather, provide access to, and report on data that is useful to business decision-makers. BI sometimes also includes the processes and technologies for data visualization and information presentation.

**Business Intelligence and Performance Management Software**


**Oracle Business Intelligence Solutions**

The First Comprehensive, Cost-Effective BI Solution Only Oracle delivers a complete, pre-integrated technology foundation to reduce the cost and complexity of deploying and maintaining BI apps. Explore BI powered by Oracle.
Money

$2,718,281,828
The target for Google's IPO on April 29, 2004. This somewhat strange number is the equivalent of the mathematical constant $e = 2.718281828$.

Revenue

<table>
<thead>
<tr>
<th>Year</th>
<th>Revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>$19,108,000</td>
</tr>
<tr>
<td>2001</td>
<td>$86,426,000</td>
</tr>
<tr>
<td>2002</td>
<td>$349,508,000</td>
</tr>
<tr>
<td>2003</td>
<td>$1,185,934,000</td>
</tr>
<tr>
<td>2004</td>
<td>$3,189,223,000</td>
</tr>
<tr>
<td>2005</td>
<td>$5,135,550,000</td>
</tr>
<tr>
<td>2006</td>
<td>$10,604,020,000</td>
</tr>
<tr>
<td>2007</td>
<td>$18,505,990,000</td>
</tr>
<tr>
<td>2008</td>
<td>$23,793,550,000</td>
</tr>
<tr>
<td>2009</td>
<td>$22,850,560,000</td>
</tr>
</tbody>
</table>

Profit

<table>
<thead>
<tr>
<th>Year</th>
<th>Profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>$1,640,000</td>
</tr>
<tr>
<td>2001</td>
<td>$5,985,000</td>
</tr>
<tr>
<td>2002</td>
<td>$96,050,000</td>
</tr>
<tr>
<td>2003</td>
<td>$105,046,000</td>
</tr>
<tr>
<td>2004</td>
<td>$299,119,000</td>
</tr>
<tr>
<td>2005</td>
<td>$1,465,397,000</td>
</tr>
<tr>
<td>2006</td>
<td>$3,077,450,000</td>
</tr>
<tr>
<td>2007</td>
<td>$4,703,720,000</td>
</tr>
<tr>
<td>2008</td>
<td>$4,225,880,000</td>
</tr>
<tr>
<td>2009</td>
<td>$6,520,450,000</td>
</tr>
</tbody>
</table>

Percent of revenue from advertising: 97%

Advertising revenue

<table>
<thead>
<tr>
<th>Year</th>
<th>Revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002</td>
<td>$460,015,000</td>
</tr>
<tr>
<td>2003</td>
<td>$1,420,663,000</td>
</tr>
<tr>
<td>2004</td>
<td>$3,143,288,000</td>
</tr>
<tr>
<td>2005</td>
<td>$6,063,002,000</td>
</tr>
<tr>
<td>2006</td>
<td>$10,492,626,000</td>
</tr>
<tr>
<td>2007</td>
<td>$16,412,043,000</td>
</tr>
<tr>
<td>2008</td>
<td>$21,126,514,000</td>
</tr>
<tr>
<td>2009</td>
<td>$22,868,804,000</td>
</tr>
</tbody>
</table>

Stock price

<table>
<thead>
<tr>
<th>Year</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>2004</td>
<td>$85</td>
</tr>
<tr>
<td>February 22</td>
<td>$535</td>
</tr>
</tbody>
</table>

Selected acquisitions

<table>
<thead>
<tr>
<th>Date</th>
<th>Company</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feb, 2003</td>
<td>Pyra Labs</td>
<td>Weblog software</td>
</tr>
<tr>
<td>Mar, 2004</td>
<td>Urchin</td>
<td>Web analytics</td>
</tr>
<tr>
<td>Aug 15, 2005</td>
<td>Android</td>
<td>Mobile software</td>
</tr>
<tr>
<td>Oct 9, 2006</td>
<td>YouTube</td>
<td>Video sharing</td>
</tr>
<tr>
<td>Apr 13, 2007</td>
<td>DoubleClick</td>
<td>Online advertising</td>
</tr>
<tr>
<td>July 9, 2007</td>
<td>Postini</td>
<td>Email security</td>
</tr>
<tr>
<td>Nov 9, 2009</td>
<td>AdMabs</td>
<td>Mobile advertising</td>
</tr>
</tbody>
</table>

Revenue by geography

- USA: 47%
- UK: 13%
- Rest of world: 40%

Assets (Dec 31, 2009)

$40.5 billion
How To Measure “Importance”

Landmark Result Paper

Survey Paper—Big Bib
How To Measure “Importance”

Landmark Result Paper

Authorities

Survey Paper—Big Bib

Hubs
How To Measure “Importance”

- Good hubs point to good authorities
- Good authorities are pointed to by good hubs
HITS

Hypertext Induced Topic Search (1998)

Determine Authority & Hub Scores

- $a_i = \text{authority score for } P_i$
- $h_i = \text{hub score for } P_i$
HITS
Hypertext Induced Topic Search (1998)

Determine Authority & Hub Scores

- $a_i = \text{authority score for } P_i$
- $h_i = \text{hub score for } P_i$

Successive Refinement

- Start with $h_i = 1$ for all pages $P_i$  \Rightarrow \ h_0 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$
HITS
Hypertext Induced Topic Search (1998)

Determine Authority & Hub Scores

• $a_i = \text{authority score for } P_i$
• $h_i = \text{hub score for } P_i$

Successive Refinement

• Start with $h_i = 1$ for all pages $P_i$ \( \Rightarrow \) $h_0 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$
• Define Authority Scores (on the first pass)

\[
a_i = \sum_{j:P_j \rightarrow P_i} h_j
\]
Determine Authority & Hub Scores

- \( a_i = \) authority score for \( P_i \)
- \( h_i = \) hub score for \( P_i \)

Successive Refinement

- Start with \( h_i = 1 \) for all pages \( P_i \) \( \Rightarrow \) \( h_0 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \)
- Define Authority Scores (on the first pass)

\[
a_i = \sum_{j:P_j \rightarrow P_i} h_j \quad \Rightarrow \quad a_1 = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = L^T h_0
\]

\[
L_{ij} = \begin{cases} 1 & P_i \rightarrow P_j \\ 0 & P_i \nrightarrow P_j \end{cases}
\]
HITS Algorithm

Refine Hub Scores

- \( h_i = \sum_{j:P_i \rightarrow P_j} a_j \quad \Rightarrow \quad h_1 = La_1 \)

\[
L_{ij} = \begin{cases} 
1 & P_i \rightarrow P_j \\
0 & P_i \not\rightarrow P_j 
\end{cases}
\]
HITS Algorithm

Refine Hub Scores
• \( h_i = \sum_{j: P_i \rightarrow P_j} a_j \Rightarrow h_1 = L a_1 \)

Successively Re-refine Authority & Hub Scores
• \( a_1 = L^T h_0 \)
HITS Algorithm

Refine Hub Scores

- $h_i = \sum_j a_j \Rightarrow h_1 = La_1$

Successively Re-refine Authority & Hub Scores

- $a_1 = L^T h_0$
  - $h_1 = La_1$

$L_{ij} = \begin{cases} 
1 & P_i \rightarrow P_j \\
0 & P_i \not\rightarrow P_j 
\end{cases}$
HITS Algorithm

Refine Hub Scores
- \( h_i = \sum_{j: P_i \rightarrow P_j} a_j \Rightarrow h_1 = La_1 \)

Successively Re-refine Authority & Hub Scores
- \( a_1 = L^T h_0 \)
  - \( h_1 = La_1 \)
  - \( a_2 = L^T h_1 \)
HITS Algorithm

Refine Hub Scores
• \( h_i = \sum_{j:P_i \to P_j} a_j \Rightarrow h_1 = La_1 \)

Successively Re-refine Authority & Hub Scores
• \( a_1 = L^T h_0 \)
• \( h_1 = La_1 \)
• \( a_2 = L^T h_1 \)
• \( h_2 = La_2 \)

\[ L_{ij} = \begin{cases} 
1 & P_i \to P_j \\
0 & P_i \not\to P_j 
\end{cases} \]
HITS Algorithm

Refine Hub Scores

• \( h_i = \sum_{j: P_i \rightarrow P_j} a_j \Rightarrow h_1 = L a_1 \)

\[ L_{ij} = \begin{cases} 1 & P_i \rightarrow P_j \\ 0 & P_i \not\rightarrow P_j \end{cases} \]

Successively Re-refine Authority & Hub Scores

• \( a_1 = L^T h_0 \)
  • \( h_1 = L a_1 \)
  • \( a_2 = L^T h_1 \)
    • \( h_2 = L a_2 \)
    ... 

Combined Iterations

• \( A = L^T L \) (authority matrix)
HITS Algorithm

Refine Hub Scores

• \( h_i = \sum_{j:P_i \rightarrow P_j} a_j \) \( \Rightarrow \) \( h_1 = L a_1 \)

\[
L_{ij} = \begin{cases} 
1 & P_i \rightarrow P_j \\
0 & P_i \not\rightarrow P_j 
\end{cases}
\]

Successively Re-refine Authority & Hub Scores

• \( a_1 = L^T h_0 \)
  • \( h_1 = L a_1 \)
  • \( a_2 = L^T h_1 \)
    • \( h_2 = L a_2 \)
    •...

Combined Iterations

• \( A = L^T L \) (authority matrix) \( a_k = A a_{k-1} \rightarrow e\)-vector (direction)
HITS Algorithm

Refine Hub Scores

\[ h_i = \sum_{j:P_i \rightarrow P_j} a_j \implies h_1 = La_1 \]

Successively Re-refine Authority & Hub Scores

\[ a_1 = L^T h_0 \]
\[ h_1 = La_1 \]
\[ a_2 = L^T h_1 \]
\[ h_2 = La_2 \]
\[ \vdots \]

Combined Iterations

\[ A = L^T L \text{ (authority matrix)} \]
\[ a_k = Aa_{k-1} \rightarrow \text{e-vector (direction)} \]

\[ H = LL^T \text{ (hub matrix)} \]
\[ h_k = Hh_{k-1} \rightarrow \text{e-vector (direction)} \]
HITS Algorithm

Refine Hub Scores

- $h_i = \sum_{j:P_i \rightarrow P_j} a_j \Rightarrow h_1 = La_1$

Successively Re-refine Authority & Hub Scores

- $a_1 = L^T h_0$
  - $h_1 = La_1$
  - $a_2 = L^T h_1$
  - $h_2 = La_2$
  
  ...  

Combined Iterations

- $A = L^T L$ (authority matrix)  
  - $a_k = Aa_{k-1} \rightarrow$ e-vector (direction)

- $H = LL^T$ (hub matrix)  
  - $h_k = Hh_{k-1} \rightarrow$ e-vector (direction)

!! A lot of work !!
Compromise

1. Do direct query matching
Compromise

1. Do direct query matching

2. Build neighborhood graph
Compromise

1. Do direct query matching
2. Build neighborhood graph

3. Compute authority & hub scores for just the neighborhood
Pros & Cons

Advantages

- Returns satisfactory results
  - Client gets both authority & hub scores
Pros & Cons

Advantages

• Returns satisfactory results
  — Client gets both authority & hub scores

Disadvantages

• Too much has to happen while client is waiting
Pros & Cons

Advantages

• Returns satisfactory results
  — Client gets both authority & hub scores

Disadvantages

• Too much has to happen while client is waiting
  — Custom built neighborhood graph needed for each query
Pros & Cons

Advantages

• Returns satisfactory results
  — Client gets both authority & hub scores

Disadvantages

• Too much has to happen while client is waiting
  — Custom built neighborhood graph needed for each query
  — Two eigenvector computations needed for each query
Pros & Cons

Advantages

• Returns satisfactory results
  — Client gets both authority & hub scores

Disadvantages

• Too much has to happen while client is waiting
  — Custom built neighborhood graph needed for each query
  — Two eigenvector computations needed for each query

• Scores can be manipulated by creating artificial hubs
HITS Applied
Every time you cough, a hum of code or a piece of some obscure url comes floating out. You can’t see it, but it’s there, invisible.

A little string of binary code, or maybe the “r” and “g” from a dot org, right there on your tongue capillary. The reason is that you’re diving into a sea of information. Need not the wondrous findings of the recent ODP coastline study—by the time glacial melt brings the ocean to your doorstep, your lungs will already be full of html.

WE DON’T HAVE TO TELL YOU THE WORLD WIDE WEB IS AN ACRONYMIC FORM OF POPULIST HYPERMEDIA.

But we will tell you it’s a hapertatous corpus of unfathomable intrigue, and it’s expanding faster than a flat universe in a cosmologically significant vacuum energy density. For the love of Goedel, just look at the thing! Billions of participants with as many agendas, cranking out hyperlinked content like there’s no tomorrow. In fact, at this rate, the disappearance of tomorrow, or at least a universally accepted definition thereof, is actually a valid concern.

SEARCH IS AN UNDERSTATEMENT.

ODYSSEUS QUEST IS MORE LIKE IT.

So how are you supposed to find anything in this great roiling mass of ones and zeros? Text-based searches are not so good. If you believe otherwise, consider the word facial. A search engine that takes nothing more than the word itself into account will return literally consistent but conceptually scattered results. On one end of the facial spectrum, there’s a mud mask. The other kind of facial, well, as anyone who rolls sans adult filter can attest, it’s a different deal altogether. Look, even if you do manage to cluster a word that has five different meanings, there’s still the fact that each individual meaning yields nearly infinite search results. And a quandary divided by five is still two hundred quandaries.

ALL OF A SUDDEN, “WHO KNOWS?” IS AN ASTUQUE QUESTION.

Searching the Internet, it turns out, is not much different from searching the real world. The best thing to do is ask someone who knows something about the subject. But who is the authority, and what qualifies them as such in the first place? A Web page can’t just declare itself an authority. If authority could be generated without the help of notable people, Louis de Branges would have verified his own proof of the Riemann Hypothesis. Yet it should authority be conferred from one page to another. This means you’d be OK setting Herman Hudspeth pick your primary care guy. Last in the triumvirate of really-bad-ways-to-determine-authority is the notion of popularity. Surprisingly, this is the method employed by today’s most widely used search engines. They find sites with the most links and present them as authorities. This is roughly analogous to handing the Fields Medal to your high school homecoming queen.

THE ANSWER CAME FROM BOOKS. WEIRD.

So what’s the solution to search? While computer science was trying to come up with an answer from its collective back drive, it was sitting right there in the stacks all along. Who could have guessed that when Eugene Garfield went all bibliometric and devised a system to find out how much a journal mattered by counting the number of times that journal was cited in other publications, he was actually inventing the beginnings of a system that might work in search. Then Gabriele Pinski and Francis Narin took it a step further by suggesting one could also rank websites, and let’s face it, being cited in the Spring ’06 issue of Social Text (pages 217–252) is inexact but certainly a literary feather in your cap. But taking into account the quality of citations is only half the answer in search.

Because compared to the neatly governed world of scientific publishing, the Internet is completely insane. Fluid. Volatile. Heterogeneous. Awaits in anonymity. Replete with conflicting agendas. So counting inbound links isn’t enough. Not even close. To search effectively in these circumstances, you have to do some serious math gaggles and take a look at the big picture.

THE ALGORITHM SEES GALAXIES, BUT IT’S BLIND AS A BAT.

The heavy hitters of search all use the same mathematically myopic approach—counting links back to authoritative Web pages. But the only way to tell what’s really going on is to take a step back and look for patterns in the sites that point back to authorities. And when you do, you quickly see that there is another layer to the puzzle—the sites that point to more than one authority, or hub pages, if you will. These hubs and their surrounding authorities form little galaxies of relevance information, something that makes the hair stand up on the back of any self-respecting searchengine’s neck. It’s the difference between checking out the Big Dipper from a lawn chair in your backyard and peering into Fornax with Hubbell’s Ultra Deep Field. But an algorithm that could detect these galaxies would be virtually impossible to pull off, since it would have to assess both inbound and outbound information, and continually calculate the relationship between the two, in real time.

THE ALGORITHM IS RELATIVELY SIMPLE, IF YOU’RE SOME KIND OF SAINT.

It works like this. For each search query, an index G of Web pages is found. For each page p, you associate a non-negative authority weight $a(p)$ and a non-negative hub weight $h(p)$. This will lead you to the rather obvious conclusion that when p points to lots of pages with big $h$ values, it should get a big $h$ value (inverse weighted popularity). And when p is pointed to by lots of pages with big $h$ values, it should get a big $h$ value (weighted popularity). From here, you simply fire up an iterative singular value decomposition operation and wrap things up by bouncing out an orthonormal basis of eigenspace for each and obtaining the eigenvectors for the matrices in question. That’s it.

IT’S A GOOD THING ROBERT FROST NEVER WROTE AN ALGORITHM.

Taking the road less traveled is fine if you’re stumbling around the New England countryside, being whimsical or whatever. But when you’re searching online, that kind of thing gets you eaten by wolves. Because dismissing where others have gone can quickly get you lost in a forest of irrelevant results. But while you’re learning from the Algorithm, the Algorithm is learning too. It studies the way anonymous groups of users search and forms an aggregate view of which results those users find the most valuable. This sends relevance through the roof and gets you to your desired destination without the slightest hint of supine intercession. Sure, “The Road Not Taken” might make a lovely poem, but it makes a gorgeous piece of code.

THE ALGORITHM APPROACHES ARTIFICIAL INTELLIGENCE, BUT IT HAS NOTHING AGAINST PEOPLE NAMED SARAH CONNOR.

Yes, the Algorithm is an omnivorous, evolving organism devoid of all feeling, but in no way should this freak you out. In fact, it’s cause for celebration. Because the Algorithm comes in peace. It’s here to revolutionize search by identifying a topic, finding experts on that topic and assessing the popularity of pages among those experts, simultaneously, in the blink of an eye, wherever you want. It’s there to narrow or expand your search based on a concept—something no other search engine can do. Never again will you be made into the perpetually updated, subject-centric world of blogs without technology that actually comprehends subjects. The Algorithm knows that Usher Syndrome is transmitted by an autosomal recessive gene, not a subwoofer. And never again will you get “results consisting merely of ten blue links, rather than the rich aggregate of images, video, conceptually related search topics and pure expert insight the Algorithm delivers.

THE ALGORITHM UNDERSTANDS THAT COLLECTIVE WISDOM IS NOT NECESSARILY COLLECTED FROM EVERYONE.

Based solely on the number of participants, the Web is undoubtedly the world’s largest source of pure wisdom. But this doesn’t mean there is wisdom inherent in every participant or every page. The Algorithm is acutely aware of this. It realizes that somewhere between James Surowiecki’s The Wisdom of Crowds and Charles Mackay’s Madness of Crowds lies the sweet spot. It sees everything but knows just what to look for: it scources the convoluted expanses of cyberspace and brings back an instantaneous convergence of wisdom collected, waiting for the day you’re ready.
The Next Frontiers

The New Age of Google

The Search Giant Has Changed Our Lives. Can Anybody Catch These Guys? By Steven Levy

PLUS: The Future of Digital Voting

Google founders Larry Page and Sergey Brin
Google’s PageRank

(Lawrence Page & Sergey Brin 1998)

The Google Goals

- Create a PageRank \( r(P) \) that is not query dependent
  - Off-line calculations — No query time computation
Google’s PageRank
(Lawrence Page & Sergey Brin 1998)

The Google Goals

- Create a PageRank $r(P)$ that is not query dependent
  - Off-line calculations — No query time computation
- Let the Web vote with in-links
Google’s PageRank

(Lawrence Page & Sergey Brin 1998)

The Google Goals

• Create a PageRank $r(P)$ that is not query dependent
  ▶ Off-line calculations — No query time computation

• Let the Web vote with in-links
  ▶ But not by simple link counts
    — One link to $P$ from Yahoo! is important
    — Many links to $P$ from me is not
Google’s PageRank
(Lawrence Page & Sergey Brin 1998)

The Google Goals

• Create a PageRank \( r(P) \) that is not query dependent
  ▶ Off-line calculations — No query time computation

• Let the Web vote with in-links
  ▶ But not by simple link counts
    — One link to \( P \) from Yahoo! is important
    — Many links to \( P \) from me is not

• Share The Vote
  ▶ Yahoo! casts many “votes”
    — value of vote from Yahoo! is diluted
  ▶ If Yahoo! “votes” for \( n \) pages
    — Then \( P \) receives only \( r(Y)/n \) credit from \( Y \)
PageRank

Google’s Original Idea

$$r(P) = \sum_{P \in B_P} \frac{r(P)}{|P|}$$

$$B_P = \{\text{all pages pointing to } P\}$$

$$|P| = \text{number of out links from } P$$
PageRank

Google’s Original Idea

\[ r(P) = \sum_{P \in B_P} \frac{r(P)}{|P|} \]

\[ B_P = \{ \text{all pages pointing to } P \} \]

\[ |P| = \text{number of out links from } P \]

Successive Refinement

Start with \( r_0(P_i) = 1/n \) for all pages \( P_1, P_2, \ldots, P_n \)
PageRank

Google’s Original Idea

\[ r(P) = \sum_{P \in \mathcal{B}_P} \frac{r(P)}{|P|} \]

\( \mathcal{B}_P = \{ \text{all pages pointing to } P \} \)

\( |P| = \text{number of out links from } P \)

Successive Refinement

Start with \( r_0(P_i) = \frac{1}{n} \) for all pages \( P_1, P_2, \ldots, P_n \)

Iteratively refine rankings for each page

\[ r_1(P_i) = \sum_{P \in \mathcal{B}_{P_i}} \frac{r_0(P)}{|P|} \]
PageRank

Google’s Original Idea

\[ r(P) = \sum_{P \in B_P} \frac{r(P)}{|P|} \]

\( B_P = \{ \text{all pages pointing to } P \} \)

\( |P| = \text{number of out links from } P \)

Successive Refinement

Start with \( r_0(P_i) = 1/n \) for all pages \( P_1, P_2, \ldots, P_n \)

Iteratively refine rankings for each page

\[ r_1(P_i) = \sum_{P \in B_{P_i}} \frac{r_0(P)}{|P|} \]

\[ r_2(P_i) = \sum_{P \in B_{P_i}} \frac{r_1(P)}{|P|} \]
PageRank

Google’s Original Idea

\[ r(P) = \sum_{P \in B_P} \frac{r(P)}{|P|} \]

\( B_P = \{ \text{all pages pointing to } P \} \)

\(|P| = \text{number of out links from } P \)

Successive Refinement

Start with \( r_0(P_i) = 1/n \) for all pages \( P_1, P_2, \ldots, P_n \)

Iteratively refine rankings for each page

\[ r_1(P_i) = \sum_{P \in B_{P_i}} \frac{r_0(P)}{|P|} \]

\[ r_2(P_i) = \sum_{P \in B_{P_i}} \frac{r_1(P)}{|P|} \]

\[ \vdots \]

\[ r_{j+1}(P_i) = \sum_{P \in B_{P_i}} \frac{r_j(P)}{|P|} \]
In Matrix Notation

After Step $k$

$$\pi_k^T = [r_k(P_1), r_k(P_2), \cdots, r_k(P_n)]$$
In Matrix Notation

After Step $k$

- $\pi_t^T = [r_k(P_1), r_k(P_2), \cdots, r_k(P_n)]$

- $\pi_{k+1}^T = \pi_k^T H$ where $h_{ij} = \begin{cases} 
1/|P_i| & \text{if } i \to j \\
0 & \text{otherwise}
\end{cases}$
In Matrix Notation

After Step $k$

\[ \pi_k^T = [r_k(P_1), r_k(P_2), \ldots, r_k(P_n)] \]

\[ \pi_{k+1}^T = \pi_k^T H \quad \text{where} \quad h_{ij} = \begin{cases} 1/|P_i| & \text{if } i \rightarrow j \\ 0 & \text{otherwise} \end{cases} \]

PageRank vector = $\pi^T = \lim_{k \to \infty} \pi_k^T = \text{eigenvector for } H$

\[ \pi^T = \pi^T H \]
$$H = \begin{pmatrix} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{pmatrix}$$
Tiny Web

\[
H = \begin{pmatrix}
    P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
    P_1 & 0 & 1/2 & 1/2 & 0 & 0 & 0 \\
    P_2 & 1/2 & 0 & 0 & 0 & 0 & 0 \\
    P_3 & 1/2 & 0 & 0 & 0 & 0 & 0 \\
    P_4 & 0 & 0 & 0 & 0 & 0 & 0 \\
    P_5 & 0 & 0 & 0 & 0 & 0 & 0 \\
    P_6 & 0 & 0 & 0 & 0 & 0 & 0 
\end{pmatrix}
\]
Tiny Web

\[
H = \begin{pmatrix}
P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
0 & 1/2 & 1/2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]
Tiny Web

\[
H = \begin{pmatrix}
P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
P_1 & 0 & 1/2 & 1/2 & 0 & 0 & 0 \\
P_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
P_3 & 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
P_4 & & & & & & \\
P_5 & & & & & & \\
P_6 & & & & & & 
\end{pmatrix}
\]
Tiny Web

$$H = \begin{pmatrix}
    P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
    P_1 & 0 & 1/2 & 1/2 & 0 & 0 & 0 \\
    P_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
    P_3 & 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
    P_4 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\
    P_5 \\
    P_6
\end{pmatrix}$$
Tiny Web

\[ H = \begin{pmatrix}
0 & 1/2 & 1/2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
0 & 0 & 0 & 0 & 1/2 & 1/2 \\
0 & 0 & 0 & 1/2 & 0 & 1/2 \\
\end{pmatrix} \]
Tiny Web

\[
H = \begin{pmatrix}
0 & 1/2 & 1/2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
0 & 0 & 0 & 0 & 1/2 & 1/2 \\
0 & 0 & 0 & 1/2 & 0 & 1/2 \\
0 & 0 & 0 & 1 & 0 & 0
\end{pmatrix}
\]
Tiny Web

H =

\[
\begin{pmatrix}
P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
P_1 & 0 & 1/2 & 1/2 & 0 & 0 & 0 \\
P_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
P_3 & 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
P_4 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\
P_5 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\
P_6 & 0 & 0 & 0 & 1 & 0 & 0 \\
\end{pmatrix}
\]

▷ A random walk on the Web Graph
A random walk on the Web Graph

PageRank = $\pi_i = \text{amount of time spent at } P_i$
A random walk on the Web Graph

PageRank = \( \pi_i = \) amount of time spent at \( P_i \)

Dead end page (nothing to click on) — a "dangling node"
A random walk on the Web Graph

PageRank = $\pi_i = \text{amount of time spent at } P_i$

Dead end page (nothing to click on) — a “dangling node”

$\pi^T = (0, 1, 0, 0, 0, 0) \implies \text{Page } P_2 \text{ is a “rank sink”}$
The Fix

Allow Web Surfers To Make Random Jumps
The Fix
Allow Web Surfers To Make Random Jumps

Replace zero rows with \( \frac{\mathbf{e}^T}{n} = \left( \frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n} \right) \)

\[
\mathbf{s} = \begin{pmatrix}
P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
P_1 & 0 & 1/2 & 1/2 & 0 & 0 & 0 \\
P_2 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\
P_3 & 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
P_4 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\
P_5 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\
P_6 & 0 & 0 & 0 & 1 & 0 & 0 \\
\end{pmatrix}
\]
Nasty Problem

The Web Graph Is Not Strongly Connected
Nasty Problem

The Web Graph Is Not Strongly Connected

- i.e., \( S \) is a reducible matrix

\[
S = \begin{pmatrix}
P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
0 & 1/2 & 1/2 & 0 & 0 & 0 \\
1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\
1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
0 & 0 & 0 & 0 & 1/2 & 1/2 \\
0 & 0 & 0 & 1/2 & 0 & 1/2 \\
0 & 0 & 0 & 1 & 0 & 0 \\
\end{pmatrix}
\]
Irreducibility Is Not Enough

Could Get Trapped Into A Cycle

\[ P_i \rightarrow P_j \rightarrow P_i \]

\[ \pi^T = (0 \cdots 1/2 \ 0 \cdots 1/2 \ 0 \cdots 0) \]

\[ ^\uparrow \quad \quad \quad \quad \quad \uparrow \quad \quad \quad \quad \quad \]

\[ i \quad \quad \quad \quad \quad j \]
The Google Fix

Allow A Random Jump From Any Page

\[ G = \alpha S + (1 - \alpha)E > 0, \quad 0 < \alpha < 1 \]  \hspace{1cm} (\alpha \approx .85)

\[ E = \frac{1}{n} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \quad \pi^T G = \pi^T \]
The Google Fix

Allow A Random Jump From Any Page

\[ G = \alpha S + (1 - \alpha)E > 0, \quad 0 < \alpha < 1 \quad (\alpha \approx 0.85) \]

\[ E = \frac{1}{n} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \]

\[ \pi^T G = \pi^T \]

\[ E = uv^T = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix} = \begin{bmatrix} u_1v_1 & u_1v_2 & \cdots & u_1v_n \\ u_2v_1 & u_2v_2 & \cdots & u_2v_n \\ \vdots & \vdots & \ddots & \vdots \\ u_nv_1 & u_nv_2 & \cdots & u_nv_n \end{bmatrix} \]
Google, Others Start to Comb Users’ Online Habits to Tailor Results to Personal Interests

By JESSICA E. VASCHELLARO And KEVIN J. DELANEY

Search engines have long generated the same results for queries whether the person searching was a mom, mathematician or movie star. Now, who you are and what you’re interested in is starting to affect the outcome of your search.

Google Inc. and a wide range of start-ups are trying to translate factors like where you live, the ads you click on and the types of restaurants you search for into more-relevant search results. A chef who searched for “beef,” for example, might be more likely to find recipes than encyclopedia entries about livestock. And a film buff who searched for a new movie might see detailed articles about the making of the film, rather than ticket-buying sites.

Google has been enhancing and more widely deploying its search-personalization technology. Within coming weeks, Google users who are logged in will begin having their search results reordered based on information they have provided to Google. For instance, they may have entered a city to receive weather forecasts on a personalized Google home page. As a result, a user in New York who types in “Giants” might see higher search results for the football team than a user in San Francisco, who might be more interested in the Giants baseball team.

Consumers who use its Web-history service to track previous search queries currently get results that are influenced by those queries and the sites they have clicked on. The company plans eventually to offer personalization based on a user’s Web-browsing history—including sites people visited without going through Google—when users agree to let Google track it.

Also, within three to five years, Google will...
Always Changing

PR Augmented With Content Scores For Final Rankings

“Metrics” Are Proprietary — But Known Examples

- Whether query terms appear in the title or the body
- Number of times query terms appear in a page
- Proximity of multiple query words to one another
- Appearance of query terms in a page (e.g., headings in bold font score higher)
- Content of neighboring web pages
Always Changing

PR Augmented With Content Scores For Final Rankings

“Metrics” Are Proprietary — But Known Examples

— Whether query terms appear in the title or the body
— Number of times query terms appear in a page
— Proximity of multiple query words to one another
— Appearance of query terms in a page (e.g., headings in bold font score higher)
— Content of neighboring web pages

Every Thursday

— Three dozen engineers, product managers, and executives make Google smarter
— This year (2010), Google plans to introduce about 550 improvements
Improvement History

🌟 Backrub [September 1997]
   - Had run on Stanford servers for almost two years—renamed Google.

🌟 New algorithm [August 2001]
   - Search algorithm completely revamped—incorporated additional ranking criteria

🌟 Local connectivity analysis [February 2003]
   - Gives more weight to links from authoritative sites

🌟 Fritz [Summer 2003]
   - Update the index constantly instead of in big batches

🌟 Personalization [June 2005]
   - Mine search behavior to provide individualized results

🌟 Bigdaddy [December 2005]
   - Engine update allows for more-comprehensive Web crawling

🌟 Universal search [May 2007]
   - Provide links to any medium (image, news, books) on the same results page

🌟 Real-Time Search [December 2009]
   - Results from Twitter and blogs as they are published
Conclusion

Google and PageRank is changing the world.

Thank you.