Information Retrieval
Web Search

Carl Meyer

Department of Mathematics
North Carolina State University
Raleigh, NC

SAC Cap Man 7/14/2005
Early Search Engines

System for the Mechanical Analysis and Retrieval of Text

Harvard 1962 – 1965

IBM 7094 & IBM 360

Gerard Salton

Implemented at Cornell (1965 – 1970)

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 $f_{ij} = \# \text{ times term } i \text{ appears in document } j$
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 \( f_{ij} = \# \text{ times term } i \text{ appears in document } j \)

Term–Document Matrix

\[
\begin{pmatrix}
\begin{array}{cccc}
\text{Doc 1} & \text{Doc 2} & \cdots & \text{Doc } n \\
\hline
\text{TERM 1} & f_{11} & f_{12} & \cdots & f_{1n} \\
\text{TERM 2} & f_{21} & f_{22} & \cdots & f_{2n} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\text{TERM m} & f_{m1} & f_{m2} & \cdots & f_{mn}
\end{array}
\end{pmatrix} = A_{m \times n}
\]
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?

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

Query Vector

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

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

How Close is Query to Each Document?

i.e., how close is \( \mathbf{q} \) to each column \( \mathbf{A}_i \)?

\[ \| \mathbf{q} - \mathbf{A}_1 \| < \| \mathbf{q} - \mathbf{A}_2 \| \text{ but } \theta_2 < \theta_1 \]
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 \( A_i \)?

\[ \|q - A_1\| < \|q - A_2\| \text{ but } \theta_2 < \theta_1 \]

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

Rank documents by size of \( \delta_i \)
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 \( A_i \)?

\[ \|q - A_1\| < \|q - A_2\| \text{ but } \theta_2 < \theta_1 \]

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 \( \delta_i \geq tol \)
Term Weighting

A Problem

Suppose query = \textit{HEDGE FUND}

If \textit{HEDGE FUND} occurs once in $D_1$ and twice in $D_2$

$\triangleright$ Then $\delta_2 \approx 2\delta_1$  \hspace{1cm} (if $\|A_1\| \approx \|A_2\|$)
Term Weighting

A Problem

Suppose query = \textit{HEDGE FUND}

If \textit{HEDGE FUND} occurs once in $D_1$ and twice in $D_2$

\[ \triangleright \text{Then } \delta_2 \approx 2\delta_1 \quad (\text{if } \|A_1\| \approx \|A_2\|) \]

To Compensate

Set $a_{ij} = \log(1 + f_{ij})$ \hfill (Other weights also used)
Term Weighting

A Problem

Suppose query = *HEDGE FUND*

If *HEDGE FUND* occurs once in $D_1$ and twice in $D_2$

- Then $\delta_2 \approx 2\delta_1$ (if $\|A_1\| \approx\|A_2\|$)

To Compensate

Set $a_{ij} = \log(1 + f_{ij})$ (Other weights also used)

Query Weighting Also Performed
Uncertainties

Ambiguity in Vocabulary

A *plane* could be ⋮
Ambiguity in Vocabulary

A *plane* could be · · ·

— A flat geometrical object
Uncertainties

Ambiguity in Vocabulary

A *plane* could be · · ·

- A flat geometrical object
- A woodworking tool
Uncertainties

Ambiguity in Vocabulary

A $plane$ could be ⋯

- A flat geometrical object
- A woodworking tool
- A Boeing product
Uncertainties

Ambiguity in Vocabulary

A *plane* could be ⋯

— A flat geometrical object
— A woodworking tool
— A Boeing product

Variation in Writing Style

No two authors write the same way
Uncertainties

Ambiguity in Vocabulary

A \textit{plane} could be \ldots

\begin{itemize}
  \item A flat geometrical object
  \item A woodworking tool
  \item A Boeing product
\end{itemize}

Variation in Writing Style

No two authors write the same way

\begin{itemize}
  \item One author may write \textit{car} and \textit{laptop}
Uncertainties

Ambiguity in Vocabulary

A \textit{plane} could be · · ·

— A flat geometrical object
— A woodworking tool
— A Boeing product

Variation in Writing Style

No two authors write the same way

— One author may write \textit{car} and \textit{laptop}
— Another author may write \textit{automobile} and \textit{portable}
Uncertainties

Ambiguity in Vocabulary

A *plane* could be · · ·

— A flat geometrical object
— A woodworking tool
— A Boeing product

Variation in Writing Style

No two authors write the same way

— One author may write *car* and *laptop*
— Another author may write *automobile* and *portable*

Variation in Indexing Conventions

— No two people index documents the same way
— Computer indexing is inexact and can be unpredictable
Theory vs Practice

In Theory — it’s simple and elegant
Theory vs Practice

In Theory — it’s simple and elegant

- Index Docs — Weight frequencies in $A$ — Normalize $\|A_i\| = 1$
Theory vs Practice

In Theory — it’s simple and elegant

- Index Docs — Weight frequencies in $A$— Normalize $\|A_i\| = 1$
- For each query, Weight terms — Normalize $\|q\| = 1$
Theory vs Practice

In Theory — it’s simple and elegant

— Index Docs — Weight frequencies in $A$ — Normalize $\|A_i\| = 1$
— For each query, Weight terms — Normalize $\|q\| = 1$
— Compute $\delta_i = \cos \theta_i = (q^T A)_i$ to return the most relevant docs
Theory vs Practice

In Theory — it’s simple and elegant

— Index Docs — Weight frequencies in $A$ — Normalize $\|A_i\| = 1$
— For each query, Weight terms — Normalize $\|q\| = 1$
— Compute $\delta_i = \cos \theta_i = (q^T A)_i$ to return the most relevant docs

In Practice — it breaks down
Theory vs Practice

In Theory — it’s simple and elegant

— Index Docs — Weight frequencies in $A$— Normalize $\|A_i\| = 1$
— For each query, Weight terms — Normalize $\|q\| = 1$
— Compute $\delta_i = \cos \theta_i = (q^T A)_i$ to return the most relevant docs

In Practice — it breaks down

— Suppose query = $car$
Theory vs Practice

In Theory — it’s simple and elegant

- Index Docs — Weight frequencies in $\mathbf{A}$— Normalize $\|\mathbf{A}_i\| = 1$
- For each query, Weight terms — Normalize $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos \theta_i = (\mathbf{q}^T \mathbf{A})_i$ to return the most relevant docs

In Practice — it breaks down

- Suppose query = car
- $D_1$ indexed by gas, car, tire

(found)
Theory vs Practice

In Theory — it’s simple and elegant

— Index Docs — Weight frequencies in \( \mathbf{A} \) — Normalize \( \| \mathbf{A}_i \| = 1 \)
— For each query, Weight terms — Normalize \( \| \mathbf{q} \| = 1 \)
— Compute \( \delta_i = \cos \theta_i = (\mathbf{q}^T \mathbf{A})_i \) to return the most relevant docs

In Practice — it breaks down

— Suppose query = \textit{car}
— \( D_1 \) indexed by \textit{gas, car, tire} (found)
— \( D_2 \) indexed by \textit{automobile, fuel, and tire} (missed)
Theory vs Practice

In Theory — it’s simple and elegant

— Index Docs — Weight frequencies in $A$— Normalize $\|A_i\| = 1$
— For each query, Weight terms — Normalize $\|q\| = 1$
— Compute $\delta_i = \cos \theta_i = (q^T A)_i$ to return the most relevant docs

In Practice — it breaks down

— Suppose query = car
— $D_1$ indexed by gas, car, tire (found)
— $D_2$ indexed by automobile, fuel, and tire (missed)

The Challenge

— Find $D_2$ by revealing the latent connection through tire
Approximate $A$ with a lower rank matrix

- Great Idea!  —>  2 patents for Bell/Telcordia


(Resource: USPTO http://patft.uspto.gov/netahtml/srchnum.htm)
Latent Semantic Indexing

Use a Fourier expansion of $A$

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

Use a Fourier expansion of $A$

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

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

Use a Fourier expansion of $A$

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

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

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

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

Use a Fourier expansion of $A$

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

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

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

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

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

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

Use a Fourier expansion of $A$

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

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

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

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 $A$

- $D_2$ forced closer to $D_1 \implies$ better chance of finding $D_2$
Latent Semantic Indexing

Use a Fourier expansion of $A$

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

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

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

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 $A$

— $D_2$ forced closer to $D_1 \implies$ better chance of finding $D_2$

“Best” mathematical solution

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

$$Z_i = u_i v_i^T$$
Pros & Cons

Cons

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

• Only semantic content used (Any link structure ignored)

• Difficult to add & delete documents

• Finding optimal $k$ not easy (Empirical tuning required)

• Doesn’t scale up well (Impractical for WWW)

• $u_i, v_i$ mixed sign $\Rightarrow$ no good interpretation

Pro

• Good at clustering $\Rightarrow$ reveals patterns for text mining
Pros & Cons

Cons

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

• Only semantic content used    (Any link structure ignored)

• Difficult to add & delete documents

• Finding optimal $k$ not easy    (Empirical tuning required)

• Doesn’t scale up well    (Impractical for WWW)

• $u_i, v_i$ mixed sign $\Rightarrow$ no good interpretation

Pro

• Good at clustering $\Rightarrow$ reveals patterns for text mining
Pros & Cons

Cons

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

- Only semantic content used   (Any link structure ignored)

- Difficult to add & delete documents

- Finding optimal $k$ not easy   (Empirical tuning required)

- Doesn’t scale up well   (Impractical for WWW)

- $u_i, v_i$ mixed sign $\Rightarrow$ no good interpretation

Pro

- Good at clustering $\Rightarrow$ reveals patterns for text mining
Pros & Cons

Cons

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

• Only semantic content used  (Any link structure ignored)

• Difficult to add & delete documents

• Finding optimal $k$ not easy  (Empirical tuning required)

• Doesn’t scale up well  (Impractical for WWW)

• $u_i, v_i$ mixed sign $\Rightarrow$ no good interpretation

Pro

• Good at clustering $\Rightarrow$ reveals patterns for text mining
Pros & Cons

Cons

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

• Only semantic content used  (Any link structure ignored)

• Difficult to add & delete documents

• Finding optimal \( k \) not easy  (Empirical tuning required)

• Doesn’t scale up well  (Impractical for WWW)

• \( u_i, v_i \) mixed sign \( \Rightarrow \) no good interpretation

Pro

• Good at clustering \( \Rightarrow \) reveals patterns for text mining
Pros & Cons

Cons

- Rankings are query dependent
  Rank of each doc is recomputed for each query
- Only semantic content used (Any link structure ignored)
- Difficult to add & delete documents
- Finding optimal $k$ not easy (Empirical tuning required)
- Doesn’t scale up well (Impractical for WWW)
- $\mathbf{u}_i, \mathbf{v}_i$ mixed sign $\Rightarrow$ no good interpretation

Pro

- Good at clustering $\Rightarrow$ reveals patterns for text mining
Pros & Cons

Cons

- Rankings are query dependent
  Rank of each doc is recomputed for each query
- Only semantic content used  (Any link structure ignored)
- Difficult to add & delete documents
- Finding optimal $k$ not easy  (Empirical tuning required)
- Doesn’t scale up well    (Impractical for WWW)
- $u_i, v_i$ mixed sign $\Rightarrow$ no good interpretation

Pro

- Good at clustering $\Rightarrow$ reveals patterns for text mining
Another Improvement (2000)

Use low-rank approximation with \textit{sparse nonnegative factors}

\[
A_{m \times n} \approx U_{m \times k} \Sigma_{k \times k} V^T_{k \times m}
\]

\[
A_{m \times n} \approx W_{m \times k} H_{k \times m}
\]
Nonnegative Matrix Factorization

Constrained Nonlinear Least Squares Problem

\[ A_{m \times n} \approx W_{m \times k} H_{k \times n} \quad \rightarrow \quad \begin{cases} \min \| A - WH \|_F^2 \\ W \geq 0, \quad H \geq 0, \quad \text{both sparse} \end{cases} \]
Nonnegative Matrix Factorization

Constrained Nonlinear Least Squares Problem

\[ A_{m \times n} \approx W_{m \times k} H_{k \times n} \Rightarrow \begin{cases} \min \| A - WH \|_F^2 \\ W \geq 0, \quad H \geq 0, \quad \text{both sparse} \end{cases} \]

\[ W_k = [w_1 | w_2 | \ldots | w_k] \quad \text{yields sparse nonnegative basis} \]

\[ \text{doc}_j \approx \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} A_j \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} \approx \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} w_1 h_{1j} + \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} w_2 h_{2j} + \cdots + \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} w_k h_{kj} \]

- Each \( w_i \) can be interpreted as a topic vector
  - Large \( \{w_{ij}, w_{ik}, \ldots w_{il}\} \Rightarrow w_i \) mostly about terms \( j, k, \ldots, l \)
  - \( h_{ij} \) indicates how much of \( \text{doc}_j \) is related to topic vector \( w_i \)
Example

(MEDLINE Amy Langville $k = 10$)

Highest Weighted Terms in Basis Vector $W_1$

1. ventricular
2. aortic
3. septal
4. left
5. defect
6. regurgitation
7. ventricle
8. valve
9. cardiac
10. pressure

Highest Weighted Terms in Basis Vector $W_2$

1. oxygen
2. flow
3. pressure
4. blood
5. cerebral
6. hypothermia
7. fluid
8. venous
9. arterial
10. perfusion

Highest Weighted Terms in Basis Vector $W_5$

1. children
2. child
3. autistic
4. speech
5. group
6. early
7. visual
8. anxiety
9. emotional
10. autism

Highest Weighted Terms in Basis Vector $W_6$

1. kidney
2. marrow
3. dna
4. cells
5. nephrectomy
6. unilateral
7. lymphocyte
8. bone
9. thymidine
10. rats
Example (cont)

\[ \text{doc}_5 \approx \begin{pmatrix} w_9 \\ \text{fatty} \\ \text{glucose} \\ \text{acids} \\ \text{ffa} \\ \text{insulin} \\ \vdots \end{pmatrix} \cdot 1.646 + \begin{pmatrix} w_6 \\ \text{kidney} \\ \text{marrow} \\ \text{dna} \\ \text{cells} \\ \text{neph.} \\ \vdots \end{pmatrix} \cdot 0.0103 + \begin{pmatrix} w_7 \\ \text{hormone} \\ \text{growth} \\ \text{hgh} \\ \text{pituitary} \\ \text{mg} \\ \vdots \end{pmatrix} \cdot 0.0045 + \cdots \]
Enron E-mail Data (1999–2001)

Fed investigation studied 15 million e-mail messages

- Over 500,000 messages made public
Enron E-mail Data (1999–2001)

Fed investigation studied 15 million e-mail messages
  • Over 500,000 messages made public

Enron’s Troubles 1999-2001
  • Problems with Dabhol Power Company (DPC) in India
Enron E-mail Data (1999–2001)

Fed investigation studied 15 million e-mail messages
  • Over 500,000 messages made public

Enron’s Troubles 1999-2001

  • Problems with Dabhol Power Company (DPC) in India
  • Deregulation of Calif. energy industry
    ▶ Rolling blackouts in the summer of 2000
    ▶ Subsequent investigations
Enron E-mail Data (1999–2001)

Fed investigation studied 15 million e-mail messages
- Over 500,000 messages made public

Enron’s Troubles 1999-2001
- Problems with Dabhol Power Company (DPC) in India
- Deregulation of Calif. energy industry
  - Rolling blackouts in the summer of 2000
    - Subsequent investigations
- Ill-fated Dynergy merger, Oct-Nov 2001
  - Revelation of Enron’s deceptive practices
    - Enron filed for bankruptcy in December 2001
# Mining 2001 E-mail

(M. Berry, Univ. Tenn)

<table>
<thead>
<tr>
<th>JAN</th>
<th>MAR</th>
<th>MAY</th>
<th>JUL</th>
<th>SEP</th>
<th>OCT</th>
<th>NOV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feb</td>
<td>Apr</td>
<td>Jun</td>
<td>Aug</td>
<td>Sep</td>
<td>Oct</td>
<td>Nov</td>
</tr>
</tbody>
</table>

## California Energy Crisis

- Jan 2
- Mar 3
- May 2
- Jul 3
- Sep 2

## Dynegy Merger / Bankruptcy

- Jan 1
- Mar 2
- May 3
- Jul 4
- Sep 3
- Nov 2

## Football (Texas / Fantasy)

- Jan 1
- Mar 2
- May 3
- Jul 4
- Sep 3
- Nov 2

## Dabhol / India

- Jan 1
- Mar 2
- May 3
- Jul 4
- Sep 3
- Nov 2
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 snippets

Index Server
Scans pages and does term indexing
Terms → Pages (similar to book index)
The Heart of a Search Engine

The Ranking Module

- Assign an importance value to each page
  - Independent of any query

- Google’s PageRank© technology distinguishes it from all competitors
The Process

query

Web Server
The Process

query

Web Server

Index Server
The Process

query → Web Server → Index Server → Doc Server
How To Measure “Importance”

Authorities

Hubs
How To Measure “Importance”

- Good hub pages point to good authority pages
How To Measure “Importance”

- Good hub pages point to good authority pages
- Good authorities are pointed to by good hubs
HITS Algorithm
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 Algorithm
Hypertext Induced Topic Search (1998)

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 \( P_i \) \( \Rightarrow \) \( h_0 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \)
HITS Algorithm

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 $P_i$ ⇒ $h_0 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$
- Define Authority Scores (first iterate)

\[ a_i = \sum_{j: P_j \rightarrow P_i} h_j \]
HITS Algorithm
Hypertext Induced Topic Search (1998)

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 (first iterate)

\[
\begin{align*}
a_i &= \sum_{j: P_j \rightarrow P_i} h_j \quad \Rightarrow \quad a_1 &= \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = L^T h_0 \\
L_{ij} &= \begin{cases} 1 & P_i \rightarrow P_j \\ 0 & P_i \not\rightarrow P_j \end{cases}
\end{align*}
\]
HITS Algorithm

Refine Hub Scores

\[ h_i = \sum_{j: P_i \rightarrow P_j} a_j \quad \Rightarrow \quad h_1 = L a_1 \]

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

Refine Hub Scores

- \( h_i = \sum_j a_j \quad \Rightarrow \quad h_1 = L a_1 \)

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

Successively Re-refine Authority & Hub Scores

- \( a_2 = L^T h_1 \)
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

- \( a_2 = L^T h_1 \)

- \( h_2 = L a_2 \)
HITS Algorithm

Refine Hub Scores

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

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

Successively Re-refine Authority & Hub Scores

- \( a_2 = L^T h_1 \)
  - \( h_2 = La_2 \)
  - \( a_3 = L^T h_2 \)
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

- \( a_2 = L^T h_1 \)
  - \( h_2 = L a_2 \)
    - \( a_3 = L^T h_2 \)
      - \( h_3 = L a_3 \)
      - \( \vdots \)
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
- \( a_2 = L^T h_1 \)
  - \( h_2 = La_2 \)
  - \( a_3 = L^T h_2 \)
  - \( h_3 = La_3 \)
  - ...

Combined Iterations
- \( A = L^T L \) (authority matrix)  \( a_k = Aa_{k-1} \)
- \( H = LL^T \) (hub matrix)  \( h_k = Hh_{k-1} \)
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
\[ a_2 = L^T h_1 \]
\[ h_2 = La_2 \]
\[ a_3 = L^T h_2 \]
\[ h_3 = La_3 \]
\[ \ldots \]

Combined Iterations
\[ A = L^T L \text{ (authority matrix)} \quad a_k = A a_{k-1} \rightarrow \text{e-vector} \quad \text{(direction)} \]
\[ H = LL^T \text{ (hub matrix)} \quad h_k = H h_{k-1} \rightarrow \text{e-vector} \quad \text{(direction)} \]
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
Pros & Cons

Advantages

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

Advantages

- Returns satisfactory results
  - Client gets both authority & hub scores
- Some flexibility for making refinements
Pros & Cons

Advantages

- Returns satisfactory results
  - Client gets both authority & hub scores
- Some flexibility for making refinements

Disadvantages

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

Advantages

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

• Some flexibility for making refinements

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
- Some flexibility for making refinements

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
• Some flexibility for making refinements

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
The Next Frontiers

The New Age of Google

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

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

● Let the Web determine importance
  ▶ 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 \( Y \)
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 determine importance
  - 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 \( Y \)
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 determine importance
  ▶ 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 \( Y \)
The Definition

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

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

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

The Definition

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

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

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

Successive Refinement

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

The Definition

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

\[ \mathcal{B}_P = \{ \text{all pages pointing to } P \} \]

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

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

The Definition

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

\[ \mathcal{B}_P = \{ \text{all pages pointing to } P \} \]

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

Successive Refinement

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

Iteratively refine rankings for each page

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

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

The Definition

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

\[ \mathcal{B}_P = \{ \text{all pages pointing to } P \} \]

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

Successive Refinement

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

Iteratively refine rankings for each page

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

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

\[ \vdots \]

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

After Step $j$

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

After Step $j$

$$\pi_j^T = [r_j(P_1), r_j(P_2), \cdots, r_j(P_n)]$$

$$\pi_{j+1}^T = \pi_j^T \mathbf{P} \quad \text{where} \quad p_{ij} = \begin{cases} 
\frac{1}{|P_i|} & \text{if } i \to j \\
0 & \text{otherwise}
\end{cases}$$
In Matrix Notation

After Step $j$

$$\pi_{j}^T = [r_j(P_1), r_j(P_2), \cdots, r_j(P_n)]$$

$$\pi_{j+1}^T = \pi_j^T P$$ where $$p_{ij} = \begin{cases} 1/|P_i| & \text{if } i \rightarrow j \\ 0 & \text{otherwise} \end{cases}$$

PageRank = \lim_{j \rightarrow \infty} \pi_j^T = \pi^T \quad (\text{provided limit exists})
In Matrix Notation

After Step $j$

\[
\pi_j^T = [r_j(P_1), r_j(P_2), \cdots, r_j(P_n)]
\]

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

PageRank = \lim_{j \to \infty} \pi_j^T = \pi^T

A Markov Chain?

If $P = [p_{ij}]$ is a stochastic matrix

\( p_{ij} \geq 0 \) and \( \sum_j p_{ij} = 1 \)
In Matrix Notation

After Step $j$

$$\pi_j^T = \left[ r_j(P_1), r_j(P_2), \cdots, r_j(P_n) \right]$$

$$\pi_{j+1}^T = \pi_j^T \mathbf{P} \quad \text{where} \quad p_{ij} = \begin{cases} \frac{1}{|P_i|} & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases}$$

PageRank = $\lim_{j \to \infty} \pi_j^T = \pi^T$ (provided limit exists)

A Markov Chain?

If $\mathbf{P} = \left[ p_{ij} \right]$ is a stochastic matrix ($p_{ij} \geq 0$ and $\sum_j p_{ij} = 1$)

Each $\pi_j^T$ is a probability vector ($\pi_i \geq 0$ and $\sum_i \pi_i = 1$)
In Matrix Notation

After Step $j$

$$\pi_j^T = [r_j(P_1), r_j(P_2), \cdots, r_j(P_n)]$$

$$\pi_{j+1}^T = \pi_j^T P \quad \text{where} \quad p_{ij} = \begin{cases} 1/|P_i| & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases}$$

PageRank = $\lim_{j \to \infty} \pi_j^T = \pi^T$ (provided limit exists)

A Markov Chain?

If $P = [p_{ij}]$ is a stochastic matrix ($p_{ij} \geq 0$ and $\sum_j p_{ij} = 1$)

Each $\pi_j^T$ is a probability vector ($\pi_i \geq 0$ and $\sum_i \pi_i = 1$)

$$\pi_{j+1}^T = \pi_j^T P$$ is random walk on the graph defined by links
In Matrix Notation

After Step $j$

$$\pi^T_j = [r_j(P_1), r_j(P_2), \cdots, r_j(P_n)]$$

$$\pi^{T}_{j+1} = \pi^{T}_j \mathbf{P} \quad \text{where} \quad p_{ij} = \begin{cases} 1/|P_i| & \text{if } i \rightarrow j \\ 0 & \text{otherwise} \end{cases}$$

PageRank = $\lim_{j \to \infty} \pi^{T}_j = \pi^T$ (provided limit exists)

A Markov Chain?

If $\mathbf{P} = [p_{ij}]$ is a stochastic matrix

Each $\pi^{T}_j$ is a probability vector

$$\pi^{T}_{j+1} = \pi^{T}_j \mathbf{P} \quad \text{is random walk on the graph defined by links}$$

$$\pi^T = \lim_{j \to \infty} \pi^{T}_j = \text{steady-state probability distribution}$$
Tiny Web

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

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

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

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

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

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

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

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

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

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

\[ \pi^T \] not well defined
The Fix

Replace zero rows with \((1/n)e^T = (1/n, 1/n, \ldots, 1/n)\)

\[
\begin{bmatrix}
P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
0 & 1/2 & 1/2 & 0 & 0 & 0 \\
1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\
1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
0 & 0 & 0 & 0 & 1/2 & 1/2 \\
0 & 0 & 0 & 1/2 & 0 & 1/2 \\
0 & 0 & 0 & 1 & 0 & 0 
\end{bmatrix}
\]
The Fix

Replace zero rows with \((1/n)e^T = (1/n, 1/n, \ldots, 1/n)\)

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

\[
S = H + \frac{ae^T}{6} = H + \begin{bmatrix}
0 \\
1 \\
0 \\
0 \\
0
\end{bmatrix} \frac{1}{6} \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix}
\]
Another Problem

S is reducible

\[ S = \begin{pmatrix}
  \begin{array}{ccc}
    P_1 & P_2 & P_3 \\
    0 & 1/2 & 1/2 \\
    1/6 & 1/6 & 1/6 \\
    1/3 & 1/3 & 0 \\
  \end{array} & \begin{array}{ccc}
    P_4 & P_5 & P_6 \\
    0 & 0 & 0 \\
    1/6 & 1/6 & 1/6 \\
    0 & 1/3 & 0 \\
  \end{array}
\end{pmatrix} \]

✓ \pi^T may not be well defined
Yet More Problems

Could get trapped into a cycle \((P_i \rightarrow P_j \rightarrow P_i)\)
Yet More Problems

Could get trapped into a cycle \((P_i \rightarrow P_j \rightarrow P_i)\)

\[\pi_{j+1}^T = \pi_j^T P \] won’t convergence
Yet More Problems

Could get trapped into a cycle \((P_i \rightarrow P_j \rightarrow P_i)\)

\[ \pi_{j+1}^T = \pi_j^T P \text{ won’t convergence} \]

Convergence Requirement

Markov chain must be irreducible and aperiodic
Yet More Problems

Could get trapped into a cycle \((P_i \rightarrow P_j \rightarrow P_i)\)

\[\pi_{j+1}^T = \pi_j^T P\] won’t convergence

Convergence Requirement

Markov chain must be irreducible and aperiodic

- This means \(P\) must be a primitive matrix

\[\checkmark\] No eigenvalues other than \(\lambda = 1\) on unit circle
Yet More Problems

**Could get trapped into a cycle** \((P_i \rightarrow P_j \rightarrow P_i)\)

\[ \pi_{j+1}^T = \pi_j^T P \] won’t convergence

**Convergence Requirement**

Markov chain must be irreducible and aperiodic

- This means \(P\) must be a primitive matrix

  \[ \pi_{j+1}^T = \pi_j^T P \] won’t convergence

  - No eigenvalues other than \(\lambda = 1\) on unit circle

  - \(P^k > 0\) for some \(k\)
Yet More Problems

Could get trapped into a cycle

\[ (P_i \rightarrow P_j \rightarrow P_i) \]

\[ \pi_T^{j+1} = \pi_T^j \mathbf{P} \text{ won’t convergence} \]

Convergence Requirement

Markov chain must be irreducible and aperiodic

- This means \( \mathbf{P} \) must be a primitive matrix
  - No eigenvalues other than \( \lambda = 1 \) on unit circle
  - \( \mathbf{P}^k > 0 \) for some \( k \)

The Google Fixes

- \( \mathbf{P} = \alpha \mathbf{S} + (1 - \alpha) \mathbf{e} \mathbf{e}^T / n \quad \alpha \approx .85 \)
Yet More Problems

Could get trapped into a cycle \((P_i \rightarrow P_j \rightarrow P_i)\)

\[
\pi_{j+1}^T = \pi_j^T P \text{ won’t convergence}
\]

Convergence Requirement

Markov chain must be irreducible and aperiodic

- This means \(P\) must be a primitive matrix
- No eigenvalues other than \(\lambda = 1\) on unit circle
- \(P^k > 0\) for some \(k\)

The Google Fixes

- \( P = \alpha S + (1 - \alpha)ee^T / n \quad \alpha \approx .85 \)
- \( P = \alpha S + (1 - \alpha)ev^T \quad v^T = \text{positive probability vector} \)
Yet More Problems

Could get trapped into a cycle \((P_i \rightarrow P_j \rightarrow P_i)\)

\[
\pi_{j+1}^T = \pi_j^T P
\]
won’t convergence

Convergence Requirement

Markov chain must be irreducible and aperiodic

- This means \(P\) must be a primitive matrix

\[
\pi_{j+1}^T = \pi_j^T P
\]

- No eigenvalues other than \(\lambda = 1\) on unit circle

\[
P^k > 0 \text{ for some } k
\]

The Google Fixes

- \(P = \alpha S + (1 - \alpha)ee^T/n\) \(\alpha \approx .85\)
- \(P = \alpha S + (1 - \alpha)e\mathbf{v}^T\) \(\mathbf{v}^T = \) positive probability vector
- \(P = \alpha H + (\alpha a + (1 - \alpha)e) \mathbf{v}^T\)
What's News—

Business and Finance

BUSH IS PREPARING to present Congress a huge bill for Iraq costs. The total could run to $95 billion depending on the length of the possible war and occupation. As horse-trading began at the U.N. to win support for a war resolution, the president again made clear he intends to act with or without the world body's imprisement. Arms inspectors said Baghdad provided new data, including a report of a possible biological bomb. Gen. Franks assumed command of the war-operations center in Qatar. Allied warplanes are aggressively taking out missile sites that could threaten the allied troop buildup, (Column 4 and Pages A4 and A5) Turkey's parliament debated legislation to let the U.S. deploy 62,000 to open a northern front. Kurdish soldiers lined roads in a show of force as U.S. officials traveled into Iraq's north for an opposition conference.

The SEC signaled it may file civil charges against Morgan Stanley, alleging it doled out IPO shares based partly on investors' commitments to buy more stock.

Ahold's problems deepened as U.S. authorities opened inquiries into accounting at the Dutch company's U.S. Foodservice unit.

Fleming said the SEC upgraded to a formal investigation an inquiry into the food wholesaler's trade practices with suppliers.

Consumer confidence fell to its lowest since 1993, hurt by energy costs, the terrorism threat and a stagnant job market.

The industrial rebounded on

World-Wide

Cat and Mouse

As Google Becomes Web's Gatekeeper, Sites Fight to Get In

Search Engine Punishes Firms That Try to Game System; Outlawing the ‘Link Farms’

Exoticleatherwear Gets Cut Off

By Michael Totty
And Mylene Mangalindan

Joy Holman sells provocative leather clothing on the Web. She wants what nearly everyone doing business online wants: more exposure on Google.

So from the time she launched exoticleatherwear.com last May, she tried all sorts of tricks to get her site to show up among the first listings when a user of Google Inc.'s popular search engine typed in “women's leatherwear” or “leather apparel.” She buried hidden words in her Web pages intended to fool Google's computers.

She signed up with a service that promised to have hundreds of sites link to her online store—thereby boosting a crucial measure in Google's system of ranking sites.

Bush to Seek up to $95 Billion To Cover Costs of War on Iraq

By Greg Jaffe
And John D. McKinnon

WASHINGTON—The Bush administration is preparing supplemental spending requests totaling as much as $95 billion for a war with Iraq, its aftermath and new expenses to fight terrorism, officials said.

The total could be as low as $60 billion because Pentagon budget planners don't know how long a military conflict will last, whether U.S. allies will contribute more than token sums to the effort and what damage Saddam Hussein might do to his own country to retaliate against conquering forces.

Budget planners also are awaiting the outcome of an intense internal debate over whether to include $15 billion in the requests to Congress that the Pentagon says it needs to fund the broader war on terrorism, as well as for stepped up homeland security. The White House Office of Management and Budget argues that the money might not be necessary. President Bush, Defense Secretary Donald Rumsfeld and budget director Mitchell Daniels Jr. met yesterday to discuss the matter but didn't reach a final agreement.
Web Sites Fight for Prime Real Estate on Google

Continued From First Page

advertising that tried to capitalize on Google's formula for ranking sites. In effect, SearchKing was offering its clients a chance to boost their own Google rankings by buying ads on more popular sites. SearchKing filed suit against the search company in federal court in Oklahoma, claiming that Google "purposely devalued" SearchKing and its customers, damaging its reputation and hurting its advertising sales.

Google won't comment on the case. In court filings, the company said SearchKing "engaged in behavior that would lower the quality of Google search results" and alter the company's ranking systems.

Google, a closely held company founded by Stanford University graduate students Sergey Brin and Larry Page, says Web companies that want to rank high should concentrate on improving their Web pages rather than gaming the system. "When people try to take scoring into their own hands, that turns into a worse experience for users," says Matt Cuts, a Google software engineer.

Coring Trickery

Efforts to outfox the search engines have been around since search engines first became popular in the early 1990s. Early tricks included stuffing thousands of widely used search terms in hidden coding, called "metatagging." The hidden coding fooled Google into identifying a site with popular words and phrases that may not actually appear on the site.

Another gimmick was hiding words or terms against a same-color background. The hidden coding deceived search engines that relied heavily on the number of times a word or phrase appeared in a ranking site. But Google's system, based on thinking, was different.

Mr. Brin, 29, one of Google's two founders and now its president of technology, boasted to a San Francisco search-engine conference in 2000 that Google wasn't worried about having its results dogged with irrelevant results because its search methods couldn't be manipulated.

That didn't stop search optimizers from finding other ways to outfox the system. Attempts to manipulate Google's results even became a sport, called Goo-
The Google Matrix

\[ P = \alpha H + (\alpha a + (1 - \alpha)e) v^T \]  
(with \( \alpha = .9 \) and \( v = e \))

\[
\begin{bmatrix}
  \frac{1}{60} & \frac{7}{15} & \frac{7}{15} & \frac{1}{60} & \frac{1}{60} & \frac{1}{60} \\
  \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\
  \frac{19}{60} & \frac{19}{60} & \frac{1}{60} & \frac{1}{60} & \frac{19}{60} & \frac{1}{60} \\
  \frac{1}{60} & \frac{1}{60} & \frac{1}{60} & \frac{1}{60} & \frac{7}{15} & \frac{7}{15} \\
  \frac{1}{60} & \frac{1}{60} & \frac{1}{60} & \frac{7}{15} & \frac{1}{60} & \frac{7}{15} \\
  \frac{1}{60} & \frac{1}{60} & \frac{1}{60} & \frac{11/12} & \frac{1}{60} & \frac{1}{60}
\end{bmatrix}
\]

The PageRank Vector

\[ \pi_{j+1}^T = \pi_j^T P \rightarrow \pi^T \]

\[
\pi^T = \begin{pmatrix}
  1 \\
  .03721 \\
  .05396 \\
  .04151 \\
  .3751 \\
  .206 \\
  .2862
\end{pmatrix}
\]
Computing $\pi^T$

A Big Problem

Solve $\pi^T = \pi^T P$ (eigenvector problem)
Computing $\pi^T$

A Big Problem

Solve $\pi^T = \pi^T P$

$\pi^T (I - P) = 0$

(eigenvector problem)

(too big for direct solves)
Google's PageRank is an eigenvector of a matrix of order 2.7 billion.

One of the reasons why Google is such an effective search engine is the PageRank™ algorithm, developed by Google's founders, Larry Page and Sergey Brin, when they were graduate students at Stanford University. PageRank is determined entirely by the link structure of the Web. It is recomputed about once a month and does not involve any of the actual content of Web pages or of any individual query. Then, for any particular query, Google finds the pages on the Web that match that query and lists those pages in the order of their PageRank.

Imagine surfing the Web, going from page to page by randomly choosing an outgoing link from one page to get to the next. This can lead to dead ends at pages with no outgoing links, or cycles around cliques of interconnected pages. So, a certain fraction of the time, simply choose a random page from anywhere on the Web. This theoretical random walk of the Web is a Markov chain or Markov process. The limiting probability that a dedicated random surfer visits any particular page is its PageRank. A page has high rank if it has links to and from other pages with high rank.

Let $W$ be the set of Web pages that can reach by following a chain of hyperlinks starting from a page at Google and let $n$ be the number of pages in $W$. The set $W$ actually varies with time, but in May 2002, $n$ was about 2.7 billion. Let $G$ be the $n$-by-$n$ connectivity matrix of $W$, that is, $G_{ij}$ is 1 if there is a hyperlink from page $i$ to page $j$ and 0 otherwise.

It tells us that the largest eigenvalue of $A$ is equal to one and that the corresponding eigenvector, which satisfies the equation

$$x = Ax,$$

exists and is unique to within a scaling factor. When this scaling factor is chosen so that

$$\sum_i x_i = 1$$

then $x$ is the state vector of the Markov chain. The elements of $x$ are Google's PageRank.

If the matrix were small enough to fit in MATLAB, one way to compute the eigenvector $x$ would be to start with a good approximate solution, such as the PageRanks from the previous month, and simply repeat the assignment statement

$$x = Ax$$

until successive vectors agree to within specified tolerance. This is known as the power method and is about the only possible approach for very large $n$. I'm not sure how Google actually computes PageRank, but one step of the power method would require one pass over a database of Web pages, updating weighted reference counts generated by the hyperlinks between pages.
Computing $\pi^T$

A Big Problem

Solve $\pi^T = \pi^T P$

$\pi^T (I - P) = 0$

Start with $\pi_0^T = e/n$ and iterate $\pi_{j+1}^T = \pi_j^T P$ (power method)
Computing $\pi^T$

A Big Problem

\begin{align*}
\text{Solve } \pi^T &= \pi^T P \\
\pi^T (I - P) &= 0 \\
\text{Start with } \pi^T_0 &= e/n \text{ and iterate } \pi^T_{j+1} &= \pi^T_j P
\end{align*}

(eigenvector problem)

(too big for direct solves)

(power method)

Convergence Time

Measured in days
Computing $\pi^T$

A Big Problem

Solve $\pi^T = \pi^T P$

$\pi^T (I - P) = 0$

Start with $\pi_0^T = e/n$ and iterate $\pi_{j+1}^T = \pi_j^T P$

Convergence Time

Measured in days

A Bigger Problem — Updating

Pages & links are added, deleted, changed continuously
Computing $\pi^T$

A Big Problem

Solve $\pi^T = \pi^T P$ \hspace{1cm} (eigenvector problem)

$\pi^T (I - P) = 0$ \hspace{1cm} (too big for direct solves)

Start with $\pi_0^T = e/n$ and iterate $\pi_{j+1}^T = \pi_j^T P$ \hspace{1cm} (power method)

Convergence Time

Measured in days

A Bigger Problem — Updating

Pages & links are added, deleted, changed continuously

Google says just start from scratch every 3 to 4 weeks
Computing $\pi^T$

A Big Problem

Solve $\pi^T = \pi^T P$

$\pi^T (I - P) = 0$

Start with $\pi_0^T = e/n$ and iterate $\pi_{j+1}^T = \pi_j^T P$ (power method)

Convergence Time

Measured in days

A Bigger Problem — Updating

Pages & links are added, deleted, changed continuously

Google says just start from scratch every 3 to 4 weeks

Prior results don’t help to restart
Conclusion

Google Now Uses Many Other “Metrics” to augment PR
Conclusion

- Google Now Uses Many Other “Metrics” to augment PR
- Elegant Blend of NA, LA, Graph Theory, Prob, & CS
Conclusion

- Google Now Uses Many Other “Metrics” to augment PR
- Elegant Blend of NA, LA, Graph Theory, Prob, & CS
- Search Is Opening New Areas Ripe For Innovative Ideas
Conclusion

- Google Now Uses Many Other “Metrics” to augment PR
- Elegant Blend of NA, LA, Graph Theory, Prob, & CS
- Search Is Opening New Areas Ripe For Innovative Ideas
- Exciting Work That Is Changing The World
Conclusion

- Google Now Uses Many Other “Metrics” to augment PR
- Elegant Blend of NA, LA, Graph Theory, Prob, & CS
- Search Is Opening New Areas Ripe For Innovative Ideas
- Exciting Work That Is Changing The World

Thanks For Your Attention