“The internet without search is like a cruise missile without a guidance system.”

Bob Davis, CEO, LYCOS
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
Robots crawl the web — software does indexing
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

Robots crawl the web — software does indexing

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

Robots crawl the web — software does indexing

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

Term–Document Matrix

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

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 \( A_i \)?

\[ \|q - A_1\| < \|q - A_2\| \text{ but } \theta_2 < \theta_1 \]
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 \( \text{query} = NCSU \)

Suppose \( NCSU \) occurs once in \( D_1 \) and twice in \( D_2 \)
Term Weighting

A Problem

Suppose query = $NCSU$

Suppose $NCSU$ occurs once in $D_1$ and twice in $D_2$

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

A Problem

Suppose query = \textit{NCSU}

Suppose \textit{NCSU} occurs once in \(D_1\) and twice in \(D_2\)

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

To Compensate

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

A Problem

Suppose query = \textit{NCSU}

Suppose \textit{NCSU} occurs once in \(D_1\) and twice in \(D_2\)

Then \(\delta_2 \approx 2\delta_1\)

To Compensate

Set \(a_{ij} = \log(1 + f_{ij})\)

(Other weights also used)

Query Weighting

Terms \textit{Boeing} and \textit{airplanes} not equally important in queries
Term Weighting

A Problem

Suppose query = \textit{NCSU}

Suppose \textit{NCSU} occurs once in \(D_1\) and twice in \(D_2\)

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

To Compensate

Set \(a_{ij} = \log(1 + f_{ij})\)

(Other weights also used)

Query Weighting

Terms \textit{Boeing} and \textit{airplanes} not equally important in queries

Importance of Term \(T_i\) in a query tends to be inversely proportional to \(\nu_i = \# \text{ Docs containing } T_i\)
Term Weighting

A Problem

Suppose query = *NCSU*

Suppose *NCSU* 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

Terms *Boeing* and *airplanes* not equally important in queries

Importance of Term $T_i$ in a query tends to be inversely proportional to $\nu_i = \#$ Docs containing $T_i$

To Compensate

Set $q_i = \begin{cases} 
\log(n/\nu_i) & \text{if } \nu_i \neq 0 \\
0 & \text{if } \nu_i = 0 
\end{cases}$ (Other weights also possible)
Uncertainties

Ambiguity in Vocabulary

A *plane* could be ⋯
Uncertainties

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 $plane$ 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 $laptop$
Uncertainties

Ambiguity in Vocabulary

A \textit{plane} could be \ldots

- 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 $\|A_i\| = 1$
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 $A$ — 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)
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
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 $A$

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

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

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

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

Use a Fourier expansion of $A$

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

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

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

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

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

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

Use a Fourier expansion of $A$

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

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

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 $A$

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

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

$$|\sigma_i| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

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

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 $A$

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

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

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 $A$

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

Possible expansions

— URV: $A = URV^T = \sum r_{ij} u_i v_j^T$
Latent Semantic Indexing

Use a Fourier expansion of A

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

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

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:\n
\[ A_k = P(A) = \sigma_1 Z_1 + \sigma_2 Z_2 + \cdots + \sigma_k Z_k \]

LSI: Query matching with \( A_k \) in place of A

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

Possible expansions

— URV: \( A = \text{URV}^T = \sum r_{ij} u_i v_j^T \)  
— SVD: \( A = \text{UDV}^T = \sum \sigma_i u_i v_j^T \)
Latent Semantic Indexing

Use a Fourier expansion of $A$

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

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

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 $D_2$

Possible expansions

— URV: $A = URV^T = \sum r_{ij} u_i v_j^T$
— SVD: $A = UDV^T = \sum \sigma_i u_i v_j^T$
— Haar: $A = H_m BH_n^T = \sum_{i,j} \beta_{ij} h_i h_j^T$ \hspace{1cm} (h’s only use -1, 0, 1)
Limitations

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

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

- Only semantic content is used
  Link structure completely ignored
Limitations

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

- Only semantic content is used
  Link structure completely ignored

- Difficult to add & delete documents
  Requires updating & downdating SVD
Limitations

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

- Only semantic content is used
  Link structure completely ignored

- Difficult to add & delete documents
  Requires updating & downdating SVD

- Determining optimal $k$ is not easy
  Empirical tuning required
Limitations

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

- Only semantic content is used
  Link structure completely ignored

- Difficult to add & delete documents
  Requires updating & downdating SVD

- Determining optimal $k$ is not easy
  Empirical tuning required

- Doesn’t scale up well
  Impractical for www
Using WWW Link Structure
Using WWW Link Structure

**Indexing**

- Still must index key terms on each page
- Robots crawl the web — software does indexing
Using WWW Link Structure

Indexing

- Still must index key terms on each page
  Robots crawl the web — software does indexing

- Inverted file structure  (like book index: terms $\rightarrow$ to pages)
  
  $Term_1 \rightarrow P_i, P_j, \ldots$
  $Term_2 \rightarrow P_k, P_l, \ldots$
  $\vdots$
  $\vdots$
Using WWW Link Structure

Indexing

- Still must index key terms on each page
  - Robots crawl the web — software does indexing

- Inverted file structure (like book index: terms $\rightarrow$ to pages)
  \[
  \begin{align*}
  &Term_1 \rightarrow P_i, P_j, \ldots \\
  &Term_2 \rightarrow P_k, P_l, \ldots \\
  &\vdots
  \end{align*}
  \]

Importance Rankings

- Attach an “importance rank” $r_i$ to each page:
  \[
  P_i \sim r_i
  \]
Using WWW Link Structure

Indexing

- Still must index key terms on each page
  Robots crawl the web — software does indexing

- Inverted file structure (like book index: terms $\rightarrow$ to pages)
  
  $\text{Term}_1 \rightarrow P_i, P_j, \ldots$
  
  $\text{Term}_2 \rightarrow P_k, P_l, \ldots$
  
  \vdots

Importance Rankings

- Attach an “importance rank” $r_i$ to each page: $P_i \sim r_i$
  
  $r_i$ based only on link structure (i.e., query independent)
Using WWW Link Structure

Indexing

- Still must index key terms on each page
  Robots crawl the web — software does indexing

- Inverted file structure (like book index: terms $\rightarrow$ to pages)
  
  $Term_1 \rightarrow P_i, P_j, \ldots$
  $Term_2 \rightarrow P_k, P_l, \ldots$
  
  \vdots

Importance Rankings

- Attach an “importance rank” $r_i$ to each page:
  
  $P_i \sim r_i$ (i.e., query independent)

  - $r_i$ based only on link structure
  - $r_i$ computed prior to any query
Using WWW Link Structure

Indexing

- Still must index key terms on each page
  Robots crawl the web — software does indexing
- Inverted file structure (like book index: terms \(\rightarrow\) to pages)
  \[\text{Term}_1 \rightarrow P_i, P_j, \ldots\]
  \[\text{Term}_2 \rightarrow P_k, P_l, \ldots\]
  \[\vdots\]

Importance Rankings

- Attach an “importance rank” \(r_i\) to each page:
  - \(r_i\) based only on link structure
  - \(r_i\) computed prior to any query
  \(P_i \sim r_i\) (i.e., query independent)

Direct Query Matching

- Query \(= (\text{Term}_1, \text{Term}_2) \rightarrow (P_i, r_i), (P_j, r_j), (P_k, r_k), \ldots\)
Using WWW Link Structure

Indexing

- Still must index key terms on each page
  Robots crawl the web — software does indexing

- Inverted file structure (like book index: terms $\rightarrow$ to pages)
  
  $Term_1 \rightarrow P_i, P_j, \ldots$
  $Term_2 \rightarrow P_k, P_l, \ldots$
  ...  

Importance Rankings

- Attach an “importance rank” $r_i$ to each page: $P_i \sim r_i$
  - $r_i$ based only on link structure (i.e., query independent)
  - $r_i$ computed prior to any query

Direct Query Matching

- Query $= (Term_1, Term_2) \rightarrow (P_i, r_i), (P_j, r_j), (P_k, r_k), \ldots$

Return $P_i, P_j, P_k, \ldots$ in order of ranks $r_i, r_j, r_k, \ldots$
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
(J. Kleinberg 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
(J. Kleinberg 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 $h_i(0) = 1$ for all pages $P_i$
HITS Algorithm
Hypertext Induced Topic Search
(J. Kleinberg 1998)

Determine Authority & Hub Scores

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

Successive Refinement

- Start with \( h_i(0) = 1 \) for all pages \( P_i \)
- Successively refine rankings
  - For \( k = 1, 2, \ldots \)
    \[
    a_i(k) = \sum_{j: P_j \rightarrow P_i} h_j(k - 1)
    \]
HITS Algorithm
Hypertext Induced Topic Search
(J. Kleinberg 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 \(h_i(0) = 1\) for all pages \(P_i\)
- Successively refine rankings
  - For \(k = 1, 2, \ldots\)
    \[a_i(k) = \sum_{j: P_j \rightarrow P_i} h_j(k - 1) \Rightarrow a_k = L^T h_{k-1}\]
HITS Algorithm
Hypertext Induced Topic Search
(J. Kleinberg 1998)

Determine Authority & Hub Scores

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

Successive Refinement

- Start with \(h_i(0) = 1\) for all pages \(P_i\)
- Successively refine rankings
  - For \(k = 1, 2, \ldots\)
    \[
    a_i(k) = \sum_{j: P_j \to P_i} h_j(k - 1) \quad \Rightarrow \quad a_k = L^T h_{k-1}
    \]
  
  \[
  h_i(k) = \sum_{j: P_i \to P_j} a_j(k)
  \]
HITS Algorithm
Hypertext Induced Topic Search
(J. Kleinberg 1998)

Determine Authority & Hub Scores

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

Successive Refinement

- Start with \( h_i(0) = 1 \) for all pages \( P_i \)
- Successively refine rankings
  - For \( k = 1, 2, \ldots \)

\[
\begin{align*}
  a_i(k) &= \sum_{j: P_j \rightarrow P_i} h_j(k - 1) \\
  h_i(k) &= \sum_{j: P_i \rightarrow P_j} a_j(k)
\end{align*}
\]

\[
\Rightarrow a_k = L^T h_{k-1}
\]

\[
\Rightarrow h_k = L a_k
\]
HITS Algorithm
Hypertext Induced Topic Search
(J. Kleinberg 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 $h_i(0) = 1$ for all pages $P_i$
- Successively refine rankings
  
  For $k = 1, 2, \ldots$
  
  $$a_i(k) = \sum_{j: P_j \rightarrow P_i} h_j(k-1) \quad \Rightarrow \quad a_k = L^T h_{k-1}$$
  $$h_i(k) = \sum_{j: P_i \rightarrow P_j} a_j(k) \quad \Rightarrow \quad h_k = La_k$$

- $A = L^T L \quad a_k = Aa_{k-1} \rightarrow \text{e-vector (direction)}$
HITS Algorithm
Hypertext Induced Topic Search (J. Kleinberg 1998)

Determine Authority & Hub Scores
- \( a_i = \) authority score for \( P_i \)
- \( h_i = \) hub score for \( P_i \)

Successive Refinement
- Start with \( h_i(0) = 1 \) for all pages \( P_i \)
- Successively refine rankings
  - For \( k = 1, 2, \ldots \)
    \[
    a_i(k) = \sum_{j: P_j \rightarrow P_i} h_j(k - 1) \quad \Rightarrow \quad a_k = L^T h_{k-1}
    \]
    \[
    h_i(k) = \sum_{j: P_i \rightarrow P_j} a_j(k) \quad \Rightarrow \quad h_k = L a_k
    \]

- \( A = L^T L \) \( a_k = A a_{k-1} \rightarrow \) e-vector (direction)
- \( H = LL^T \) \( h_k = H h_{k-1} \rightarrow \) e-vector (direction)
Compromise

1. Do direct query matching
Compromise

1. Do direct query matching
2. Build neighborhood graph
1. Do direct query matching
2. Build neighborhood graph
3. Compute authority & hub scores for just the neighborhood
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’s PageRank

(Lawrence Page & Sergey Brin 1998)

PageRank $r(P)$ Is Not Query Dependent
Google’s PageRank

(PageRank $r(P)$ Is Not Query Dependent)

- Depends primarily on link structure of web
PageRank $r(P)$ Is Not Query Dependent

- Depends primarily on link structure of web
  - Off-line calculations
  - No computation at query time
Google’s PageRank

(Lawrence Page & Sergey Brin 1998)

PageRank $r(P)$ Is Not Query Dependent

- Depends primarily on link structure of web
  - Off-line calculations
  - No computation at query time

$r(P)$ Depends On Ranks Of Pages Pointing To $P$

- Importance is not number of in-links or out-links
Google’s PageRank

(PageRank $r(P)$ is not query dependent)

- Depends primarily on link structure of web
  - Off-line calculations
  - No computation at query time

$r(P)$ depends on ranks of pages pointing to $P$

- Importance is not number of in-links or out-links
  - One link to $P$ from Yahoo! is important
Google’s PageRank

(Lawrence Page & Sergey Brin 1998)

PageRank $r(P)$ Is Not Query Dependent

- Depends primarily on link structure of web
  - Off-line calculations
  - No computation at query time

$r(P)$ Depends On Ranks Of Pages Pointing To $P$

- Importance is not number of in-links or out-links
  - One link to $P$ from Yahoo! is important
  - Many links to $P$ from me is not
Google’s PageRank
(Lawrence Page & Sergey Brin 1998)

PageRank $r(P)$ Is Not Query Dependent

- Depends primarily on link structure of web
  - Off-line calculations
  - No computation at query time

$r(P)$ Depends On Ranks Of Pages Pointing To $P$

- Importance is not number of in-links or out-links
  - One link to $P$ from Yahoo! is important
  - Many links to $P$ from me is not

PageRank Shares The Vote

- Yahoo! casts many “votes” $\Rightarrow$ value of vote from $Y$ is diluted
Google’s PageRank

(Lawrence Page & Sergey Brin 1998)

PageRank $r(P)$ Is Not Query Dependent

- Depends primarily on link structure of web
  - Off-line calculations
  - No computation at query time

$r(P)$ Depends On Ranks Of Pages Pointing To $P$

- Importance is not number of in-links or out-links
  - One link to $P$ from Yahoo! is important
  - Many links to $P$ from me is not

PageRank Shares The Vote

- Yahoo! casts many “votes” $\implies$ value of vote from $Y$ is diluted
  - If Yahoo! “votes” for $n$ pages
    - then $P$ receives only $r(Y)/n$ credit from $Y$
PageRank

The Definition

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

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

\[ |P| = \text{number of out links from } P \]
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 \( P_1, P_2, \ldots, P_n \)
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 \( P_1, P_2, \ldots, P_n \)

Iteratively refine rankings for each page
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 \( P_1, P_2, \ldots, P_n \)

Iteratively refine rankings for each page

\[ r_1(P_i) = \sum_{P \in B_{P_i}} \frac{r_0(P)}{|P|} \]
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 \( P_1, P_2, \ldots, P_n \)

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

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 \( P_1, P_2, \ldots, P_n \)

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 $j$

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

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

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

After Step $j$

$$\pi_T^j = [r_j(P_1), r_j(P_2), \cdots, r_j(P_n)]$$

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

PageRank $= \lim_{j \to \infty} \pi_T^j = \pi^T$ (provided limit exists)
In Matrix Notation

After Step $j$

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

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

where

$$p_{ij} = \begin{cases} 
1/|P_i| & \text{if } i \rightarrow j \\
0 & \text{otherwise}
\end{cases}$$

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

It’s A Markov Chain

$$P = [p_{ij}]$$ is a stochastic matrix (set $p_{ii}=1$ when all other $p_{ij}=0$)
In Matrix Notation

After Step \( j \)

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

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

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

It’s A Markov Chain

\( P = \begin{bmatrix} p_{ij} \end{bmatrix} \) is a stochastic matrix (set \( p_{ii}=1 \) when all other \( p_{ij}=0 \))

Each \( \pi_j^T \) is a probability distribution vector \( \left( \sum_i r_j(P_i)=1 \right) \)
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)

It’s A Markov Chain

\( P = [p_{ij}] \) is a stochastic matrix (set \( p_{ii}=1 \) when all other \( p_{ij}=0 \))

Each \( \pi_j^T \) is a probability distribution vector \( \left( \sum_i r_j(P_i)=1 \right) \)

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

After Step $j$

$$\pi_T^j = [r_j(P_1), r_j(P_2), \cdots, r_j(P_n)]$$

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

PageRank = $\lim_{j \rightarrow \infty} \pi_T^j = \pi_T$ (provided limit exists)

It’s A Markov Chain

$P = [p_{ij}]$ is a stochastic matrix (set $p_{ii}=1$ when all other $p_{ij}=0$)

Each $\pi_T^j$ is a probability distribution vector

$$\pi_T^{j+1} = \pi_T^j P \quad \text{is random walk on the graph defined by links}$$

$$\pi_T = \lim_{j \rightarrow \infty} \pi_T^j = \text{steady-state probability distribution}$$
Random Surfer

Web Surfer Randomly Clicks On Links

Long-run proportion of time on page $P_i$ is $\pi_i$
Random Surfer

Web Surfer Randomly Clicks On Links

Long-run proportion of time on page $P_i$ is $\pi_i$

Problems

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

Web Surfer Randomly Clicks On Links (Back button not a link)

Long-run proportion of time on page $P_i$ is $\pi_i$

Problems

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

✓ $\pi^T$ not well defined
Random Surfer

Web Surfer Randomly Clicks On Links

Long-run proportion of time on page $P_i$ is $\pi_i$

Problems

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

$\pi^T$ not well defined

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

Web Surfer Randomly Clicks On Links

Long-run proportion of time on page $P_i$ is $\pi_i$

Problems

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

$\pi^T$ not well defined

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

No convergence
Random Surfer

Web Surfer Randomly Clicks On Links (Back button not a link)
Long-run proportion of time on page $P_i$ is $\pi_i$

Problems
Dead end page (nothing to click on) — a “dangling node”
$\pi^T$ not well defined
Could get trapped into a cycle $(P_i \rightarrow P_j \rightarrow P_i)$
No convergence

Convergence
Markov chain must be irreducible and aperiodic
Random Surfer

Web Surfer Randomly Clicks On Links

Long-run proportion of time on page $P_i$ is $\pi_i$

Problems

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

$\pi^T$ not well defined

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

No convergence

Convergence

Markov chain must be irreducible and aperiodic

Bored Surfer Enters Random URL
Random Surfer

Web Surfer Randomly Clicks On Links

Long-run proportion of time on page $P_i$ is $\pi_i$.

Problems

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

$\pi^T$ not well defined

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

No convergence

Convergence

Markov chain must be irreducible and aperiodic

Bored Surfer Enters Random URL

Replace $P$ by $\tilde{P} = \alpha P + (1 - \alpha)E$ 

$e_{ij} = 1/n$ 

$\alpha \approx .85$
Random Surfer

Web Surfer Randomly Clicks On Links

Long-run proportion of time on page $P_i$ is $\pi_i$

Problems

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

  - $\pi^T$ not well defined

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

  - No convergence

Convergence

Markov chain must be irreducible and aperiodic

Bored Surfer Enters Random URL

Replace $P$ by $\tilde{P} = \alpha P + (1 - \alpha)E$  
$e_{ij} = 1/n$  
$\alpha \approx .85$

Different $E = ev^T$ and $\alpha$ allow customization & speedup
What's News—

Business and Finance

NEWS CORP. and Liberty are no longer working together on a joint offer to take control of Hughes, with News Corp. proceeding on its own and Liberty considering an independent bid. The move threatens to cloud the process of finding a new owner for the GM unit. (Article on Page A3)

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. (Article on Page C1)

Ahool'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 into the brewing of the trade practices with suppliers. (Article on Page A2)

Consumer confidence fell to its lowest since 1993, hurt by energy costs, the terrorism threat and a stagnant job market. (Page A3)

The industrial rebounded on

World-Wide

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 imprisom. Arms inspectors said Baghdad provided new data, including a report of a possible biological bomb. Gen. Franks assumed command of the war-opertations center in Qatar. Allied warplanes are aggressively taking out missile sites that could threaten the allied troop build-up. (Columns 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.

Powell said North Korea hasn't restarted a reactor and plutonium-processing facility at Yongbyon, hinting that forbearance might constitute an overture. But saber rattling continued a day after a missile test timed for the inauguration in Seoul. Pyongyang accused U.S. spy planes of violating its airspace and told its army to prepare for U.S. attack. (Page A11)

The FBI came under withering bipartisanship in a Senate judiciary hearing on the investigation of Oklahoma City. (Page A3)

Web Master

As the Web spreads...

<table>
<thead>
<tr>
<th>Total Internet users, by household, in millions</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>50</td>
</tr>
</tbody>
</table>

Google's U.S. presence expands

<table>
<thead>
<tr>
<th>Top search engines, in millions of unique visitors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
</tr>
<tr>
<td>Yahoo</td>
</tr>
<tr>
<td>MSN</td>
</tr>
<tr>
<td>AOL</td>
</tr>
<tr>
<td>Ask Jeeves</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Top shopping-referral sites, in millions of referrals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
</tr>
<tr>
<td>DealTime</td>
</tr>
<tr>
<td>BizRate</td>
</tr>
<tr>
<td>Overture</td>
</tr>
<tr>
<td>Epinions</td>
</tr>
<tr>
<td>CNET</td>
</tr>
</tbody>
</table>

*Number of people the sites send to major online stores, including only visitors from home, for Q4 2002.

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 $13 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. Mr. Rumsfeld plans to report impending billions to the White House Office of Management and Budget.

The techniques

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 MANGALDAN

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.

The techniques
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 “purposefully declined” 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 system.

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 its system. “When people try to take scoring into their own hands, that turns into a worse experience for users,” says Matt Cutts, a Google software engineer.

Coding 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 code, called “metatags.” The coding can fool search engines 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 relevance, is 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 dangled 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 Googling. One popular approach was to create Web pages that were nothing more than collections of links to the clients’ sites, called “link farms.” Since Google ranks a site largely by how many links or “votes” it gets, the link farms could boost a site’s popularity.

In a similar technique, called a link exchange, a group of unrelated sites would agree to all link to each other, thereby fooling Google into thinking the sites have a multitude of votes. Many sites also found they could buy links to themselves to boost their rankings.

Ms. Holman, the leatherwear retailer, discovered the consequences of trying to fool Google. The 42-year-old hospital laboratory technician, who learned computer skills by troubleshooting her hospital’s equipment, operates her online apparel store as a side business that she hopes can someday replace her day job.

When she launched her Exotic Leather Wear store from her home in Las Vegas, she quickly learned the importance of appearing near the top of search-engine results, especially on Google. She honed up on search techniques, visiting online discussion groups dedicated to search engines and reading what material she could find on the Web.

At first, Ms. Holman limited herself to modest changes, such as loading her page with hidden metatag coding that would encourage search engines to send traffic toward her site when a user entered words such as “halter tops” or “leather miniskirts.” Since Google doesn’t give much weight to metatags in determining its rankings, the efforts had little effect on her search results.

She then received an e-mail advertisement from AutomatedLinks.com, a Wirral, England, company that promised to send traffic “through the roof” by linking more than 2,000 Web sites to hers. Aside from attracting customers, the links were designed to improve her overall visibility by creating a search engine page that is titled “Exotic Leather Wear” so that when a user types “leather” into a search engine, the site description will appear.

In theory, when Google encounters the AutomatedLinks code, it treats it as legitimate referral traffic to the site and other sites and counting them in totaling up the site’s popularity.

Shortly after Ms. Holman signed up with AutomatedLinks.com, she read on an online discussion group that Google had bought AutomatedLinks.com. She says she immediately stripped the code from her Web pages. For a while her site gradually worked its way up in Google search results, and business steadily improved because links to her site remained on the site of other AutomatedLinks customers. Then, sometime in November, her site was suddenly no longer appearing among the top results. Home sales plunged as much as 80%.

Ms. Holman, who e-mailed Google and AutomatedLinks.com, says she has been unable to get an answer. But in the last few months, other AutomatedLinks customers say they have seen their sites apparently penalized by Google. Graham McLeay, who owns a small chauffeur service north of London, saw revenue cut in half during the two months he believes his site was penalized by Google.

The high-stakes fight between Google and the optimizers can leave some Web-site owners confused. "I don't know how people are supposed to judge what is right and wrong," says Mr. McLeay.

AutomatedLinks didn’t respond to requests for comment. Google decided to comment on the case. Mr. Cutts, the Google engineer, warns that the rules are clear and that it’s better to follow them rather than try to get a problem fixed after a site has been penalized. "We want to return the most relevant pages we can," Mr. Cutts says. "The best way for a site owner to do that is follow our guidelines.

Crackdown

Google has been stepping up its enforcement since 2001. It warned Webmasters using that trickery could get their sites kicked out of the Google index and it provided a list of forbidden activities, including hiding text and "link schemes," such as the link farms. Google also warned against "cloaking"—showing a search engine a page that’s designed to score well while giving visitors a different, more attractive page or creating multiple Web addresses that take web site visitors to a single site.

A month ago, a popular search engine added a new clause to its terms of use: "You agree not to use automated, programmatical or other means to generate traffic to your site on a large scale without our permission." The clause was added after automated links had been used to defraud Google and other search engines.

Google is working on a new system that will try to separate duplicitous sites from legitimate ones. But the changes will not be enough to satisfy competitors, who say the company is too big to ever be fair.

In theory, when Google encounters the AutomatedLinks code, it treats it as a legitimate referral to the other sites and counting them in totaling up the site’s popularity.

Shortly after Ms. Holman signed up with AutomatedLinks.com, she read on an online discussion group that Google had bought AutomatedLinks.com. She says she immediately stripped the code from her Web pages. For a while her site gradually worked its way up in Google search results, and business steadily improved because links to her site remained on the site of other AutomatedLinks customers. Then, sometime in November, her site was suddenly no longer appearing among the top results. Home sales plunged as much as 80%.

Ms. Holman, who e-mailed Google and AutomatedLinks.com, says she has been unable to get an answer. But in the last few months, other AutomatedLinks customers say they have seen their sites apparently penalized by Google. Graham McLeay, who owns a small chauffeur service north of London, saw revenue cut in half during the two months he believes his site was penalized by Google.

The high-stakes fight between Google and the optimizers can leave some Web-site owners confused. "I don't know how people are supposed to judge what is right and wrong," says Mr. McLeay.

AutomatedLinks didn’t respond to requests for comment. Google decided to comment on the case. Mr. Cutts, the Google engineer, warns that the rules are clear and that it’s better to follow them rather than try to get a problem fixed after a site has been penalized. "We want to return the most relevant pages we can," Mr. Cutts says. "The best way for a site owner to do that is follow our guidelines.

Crackdown

Google has been stepping up its enforcement since 2001. It warned Webmasters using that trickery could get their sites kicked out of the Google index and it provided a list of forbidden activities, including hiding text and "link schemes," such as the link farms. Google also warned against "cloaking"—showing a search engine a page that’s designed to score well while giving visitors a different, more attractive page or creating multiple Web addresses that take web site visitors to a single site.

A month ago, a popular search engine added a new clause to its terms of use: "You agree not to use automated, programmatical or other means to generate traffic to your site on a large scale without our permission." The clause was added after automated links had been used to defraud Google and other search engines.

Google is working on a new system that will try to separate duplicitous sites from legitimate ones. But the changes will not be enough to satisfy competitors, who say the company is too big to ever be fair.

In theory, when Google encounters the AutomatedLinks code, it treats it as a legitimate referral to the other sites and counting them in totaling up the site’s popularity.

Shortly after Ms. Holman signed up with AutomatedLinks.com, she read on an online discussion group that Google had bought AutomatedLinks.com. She says she immediately stripped the code from her Web pages. For a while her site gradually worked its way up in Google search results, and business steadily improved because links to her site remained on the site of other AutomatedLinks customers. Then, sometime in November, her site was suddenly no longer appearing among the top results. Home sales plunged as much as 80%.

Ms. Holman, who e-mailed Google and AutomatedLinks.com, says she has been unable to get an answer. But in the last few months, other AutomatedLinks customers say they have seen their sites apparently penalized by Google. Graham McLeay, who owns a small chauffeur service north of London, saw revenue cut in half during the two months he believes his site was penalized by Google.

The high-stakes fight between Google and the optimizers can leave some Web-site owners confused. "I don't know how people are supposed to judge what is right and wrong," says Mr. McLeay.

AutomatedLinks didn’t respond to requests for comment. Google decided to comment on the case. Mr. Cutts, the Google engineer, warns that the rules are clear and that it’s better to follow them rather than try to get a problem fixed after a site has been penalized. "We want to return the most relevant pages we can," Mr. Cutts says. "The best way for a site owner to do that is follow our guidelines.

Fiat Patriarch Is Set to Be Beaten

By ALESSANDRA GAZZARRINI

ROME—Umberto Agnelli, 80, who named Fiat SpA chairman 16 years ago and dominated the global auto industry for most of its profitable years, is facing a challenge to his throne.

Mr. Agnelli, the 85-year-old Fiat patriarch Gianni Agnelli, who last month, was widely expected over from current chairman Carlo de Mistico, later this year. Mr. Agnelli, who has served as chairman for over 60 years, is the elder of the Agnelli clan, which has wielded immense influence in Italy.

Fiat SpA's share price rose 65 cents to 817 lire.

"The big search engines determine the laws of how commerce runs," says Mr. Massa.
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 reached 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 $\pi_{j+1}^T = \pi_j^T P$

(eigenvector problem)

(too big for direct solves)

(power method)
Computing $\pi^T$

A Big Problem

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

$\pi^T (I - P) = 0$ (too big for direct solves)

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

Convergence Time

Measured in days
Computing $\pi^T$

A Big Problem

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

$\pi^T (I - P) = 0$ (too big for direct solves)

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
Computing $\pi^T$

A Big Problem

Solve $\pi^T = \pi^T P$  
$\pi^T (I - P) = 0$  
Start with $\pi^T_0 = e/n$ and iterate $\pi^T_{j+1} = \pi^T_j P$ (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
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

Google says just start from scratch every 3 to 4 weeks

Prior results don’t help to restart
Perron Complementation

Perron Frobenius

\[ P \geq 0, \text{ irreducible} \implies \rho(P) = \rho \in \sigma(P) \quad \text{(simple)} \]
Perron Complementation

**Perron Frobenius**

\[ P \geq 0, \text{ irreducible} \implies \rho(P) = \rho \in \sigma(P) \quad \text{(simple)} \]

**Unique Left-Hand Perron Vector**

\[ \pi^T P = \rho \pi^T \quad \pi^T > 0 \quad \| \pi^T \|_1 = 1 \]
Perron Complementation

**Perron Frobenius**
\[ P \geq 0, \text{ irreducible} \iff \rho(P) = \rho \in \sigma(P) \text{ (simple)} \]

**Unique Left-Hand Perron Vector**
\[ \pi^T P = \rho \pi^T \quad \pi^T > 0 \quad ||\pi^T||_1 = 1 \]

**Partition & Aggregate**
\[ P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \]
Perron Complementation

Perron Frobenius
\[ P \geq 0, \text{ irreducible} \implies \rho(P) = \rho \in \sigma(P) \quad \text{(simple)} \]

Unique Left-Hand Perron Vector
\[ \pi^T P = \rho \pi^T \quad \pi^T > 0 \quad \| \pi^T \|_1 = 1 \]

Partition & Aggregate
\[ P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \]

Shift \( P \) by \( \rho \)
Perron Complementation

Perron Frobenius
\( P \geq 0, \text{ irreducible} \implies \rho(P) = \rho \in \sigma(P) \) (simple)

Unique Left-Hand Perron Vector
\( \pi^T P = \rho \pi^T \quad \pi^T > 0 \quad ||\pi^T||_1 = 1 \)

Partition & Aggregate
\[ P = \begin{bmatrix}
P_{11} & P_{12} \\
P_{21} & P_{22}
\end{bmatrix} \]

Shift \( P \) by \( \rho \) \implies \text{Schur Complements}
Perron Complementation

**Perron Frobenius**

\[ P \geq 0, \text{ irreducible} \quad \implies \quad \rho(P) = \rho \in \sigma(P) \quad \text{(simple)} \]

**Unique Left-Hand Perron Vector**

\[ \pi^T P = \rho \pi^T \quad \pi^T > 0 \quad \| \pi^T \|_1 = 1 \]

**Partition & Aggregate**

\[ P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \]

Shift \( P \) by \( \rho \) \quad \rightarrow \quad \text{Schur Complements} \quad \rightarrow \quad \text{Shift back by} \ \rho
**Perron Complementation**

**Perron Frobenius**

\[ P \geq 0, \text{ irreducible} \implies \rho(P) = \rho \in \sigma(P) \quad \text{(simple)} \]

**Unique Left-Hand Perron Vector**

\[ \pi^T P = \rho \pi^T \quad \pi^T > 0 \quad \|\pi^T\|_1 = 1 \]

**Partition & Aggregate**

\[
P = \begin{bmatrix}
P_{11} & P_{12} \\
P_{21} & P_{22}
\end{bmatrix}
\]

**Shift \( P \) by \( \rho \)**

\[ \text{Schur Complements} \quad \text{Shift back by \( \rho \)} \]

**Perron Complements**

\[
S_1 = P_{11} + P_{12}(\rho I - P_{22})^{-1}P_{21} \quad S_2 = P_{22} + P_{21}(\rho I - P_{11})^{-1}P_{12}
\]
Perron Complementation

**Perron Frobenius**

\[ \mathbf{P} \geq 0, \text{ irreducible} \quad \Rightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad (\text{simple}) \]

**Unique Left-Hand Perron Vector**

\[ \pi^T \mathbf{P} = \rho \pi^T \quad \pi^T > 0 \quad \| \pi^T \|_1 = 1 \]

**Partition & Aggregate**

\[ \mathbf{P} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{bmatrix} \]

Shift \( \mathbf{P} \) by \( \rho \) \quad \rightarrow \quad \text{Schur Complements} \quad \rightarrow \quad \text{Shift back by } \rho

**Perron Complements**

\[ \mathbf{S}_1 = \mathbf{P}_{11} + \mathbf{P}_{12} (\rho \mathbf{I} - \mathbf{P}_{22})^{-1} \mathbf{P}_{21} \quad \mathbf{S}_2 = \mathbf{P}_{22} + \mathbf{P}_{21} (\rho \mathbf{I} - \mathbf{P}_{11})^{-1} \mathbf{P}_{12} \]

**Inherited Properties**

\[ \mathbf{S}_i \geq 0 \]
Perron Complementation

**Perron Frobenius**
\[ P \geq 0, \text{ irreducible} \quad \implies \quad \rho(P) = \rho \in \sigma(P) \quad \text{(simple)} \]

**Unique Left-Hand Perron Vector**
\[ \pi^T P = \rho \pi^T \quad \pi^T > 0 \quad \| \pi^T \|_1 = 1 \]

**Partition & Aggregate**
\[ P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \]

Shift \( P \) by \( \rho \)  \quad \rightarrow \quad \text{Schur Complements}  \quad \rightarrow \quad \text{Shift back by } \rho

**Perron Complements**
\[ S_1 = P_{11} + P_{12}(\rho I - P_{22})^{-1}P_{21} \quad S_2 = P_{22} + P_{21}(\rho I - P_{11})^{-1}P_{12} \]

**Inherited Properties**
\[ S_i \geq 0 \]
\[ S_i \text{ is irreducible} \]
Perron Complementation

**Perron Frobenius**

\[ \mathbf{P} \succcurlyeq 0, \text{ irreducible} \implies \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)} \]

**Unique Left-Hand Perron Vector**

\[ \pi^T \mathbf{P} = \rho \pi^T \quad \pi^T > 0 \quad \|\pi^T\|_1 = 1 \]

**Partition & Aggregate**

\[ \mathbf{P} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{bmatrix} \]

Shift \( \mathbf{P} \) by \( \rho \) \quad \Rightarrow \quad \text{Schur Complements} \quad \Rightarrow \quad \text{Shift back by} \ \rho

**Perron Complements**

\[ \mathbf{S}_1 = \mathbf{P}_{11} + \mathbf{P}_{12} (\rho \mathbf{I} - \mathbf{P}_{22})^{-1} \mathbf{P}_{21} \quad \mathbf{S}_2 = \mathbf{P}_{22} + \mathbf{P}_{21} (\rho \mathbf{I} - \mathbf{P}_{11})^{-1} \mathbf{P}_{12} \]

**Inherited Properties**

\[ \mathbf{S}_i \succcurlyeq 0 \]

\[ \mathbf{S}_i \text{ is irreducible} \]

\[ \rho(\mathbf{S}_i) = \rho = \rho(\mathbf{P}) \]
**Exact Aggregation**

**Aggregation Matrix**

\[ s_i^T = \text{Left-hand Perron vector for } S_i \]

\[ A = \begin{bmatrix} s_1^T S_1 e & s_1^T S_2 e \\ s_2^T S_1 e & s_2^T S_2 e \end{bmatrix}_{2 \times 2} \]
Exact Aggregation

Aggregation Matrix

\[ s_{i}^{T} = \text{Left-hand Perron vector for } S_{i} \]

\[ A = \begin{bmatrix} s_{1}^{T} S_{1} e & s_{1}^{T} S_{2} e \\ s_{2}^{T} S_{1} e & s_{2}^{T} S_{2} e \end{bmatrix}_{2 \times 2} \]

Inherited Properties

\[ A \geq 0 \]
Exact Aggregation

**Aggregation Matrix**

$$s_i^T = \text{Left-hand Perron vector for } S_i$$

$$A = \begin{bmatrix}
    s_1^T S_1 e & s_1^T S_2 e \\
    s_2^T S_1 e & s_2^T S_2 e
\end{bmatrix}_{2 \times 2}$$

**Inherited Properties**

$$A \geq 0$$

$$A \text{ is irreducible}$$
Exact Aggregation

Aggregation Matrix

\[ s_i^T = \text{Left-hand Perron vector for } S_i \]

\[
A = \begin{bmatrix}
    s_1^T S_1 e & s_1^T S_2 e \\
    s_2^T S_1 e & s_2^T S_2 e
\end{bmatrix}_{2 \times 2}
\]

Inherited Properties

\[ A \geq 0 \]

\[ A \text{ is irreducible} \]

\[ \rho(A) = \rho = \rho(P) = \rho(S_i) \]
Exact Aggregation

**Aggregation Matrix**

\[ \mathbf{s}_i^T = \text{Left-hand Perron vector for } \mathbf{S}_i \]

\[ \mathbf{A} = \begin{bmatrix} \mathbf{s}_1^T \mathbf{S}_1 \mathbf{e} & \mathbf{s}_1^T \mathbf{S}_2 \mathbf{e} \\ \mathbf{s}_2^T \mathbf{S}_1 \mathbf{e} & \mathbf{s}_2^T \mathbf{S}_2 \mathbf{e} \end{bmatrix}_{2 \times 2} \]

**Inherited Properties**

\[ \mathbf{A} \geq 0 \]

\[ \mathbf{A} \text{ is irreducible} \]

\[ \rho(\mathbf{A}) = \rho = \rho(\mathbf{P}) = \rho(\mathbf{S}_i) \]

**The Aggregation/Disaggregation Theorem**

Left-hand Perron vector for \( \mathbf{A} = (\alpha_1, \alpha_2) \)

\[ \rightarrow \]

Left-hand Perron vector for \( \mathbf{P} = (\alpha_1 \mathbf{s}_1^T \mid \alpha_2 \mathbf{s}_2^T) \)
Updating By Aggregation

Prior Data

\[ Q_{m \times m} = \text{Old Google Matrix} \quad \text{(known)} \]

\[ \phi^T = (\phi_1, \phi_2, \ldots, \phi_m) = \text{Old PageRank Vector} \quad \text{(known)} \]
Updating By Aggregation

Prior Data

\[ Q_{m \times m} = \text{Old Google Matrix} \quad \text{(known)} \]
\[ \phi^T = (\phi_1, \phi_2, \ldots, \phi_m) = \text{Old PageRank Vector} \quad \text{(known)} \]

Updated Data

\[ P_{n \times n} = \text{New Google Matrix} \quad \text{(known)} \]
\[ \pi^T = (\pi_1, \pi_2, \ldots, \pi_n) = \text{New PageRank Vector} \quad \text{(unknown)} \]
Updating By Aggregation

Prior Data

\[ Q_{m \times m} = \text{Old Google Matrix} \quad (\text{known}) \]
\[ \phi^T = (\phi_1, \phi_2, \ldots, \phi_m) = \text{Old PageRank Vector} \quad (\text{known}) \]

Updated Data

\[ P_{n \times n} = \text{New Google Matrix} \quad (\text{known}) \]
\[ \pi^T = (\pi_1, \pi_2, \ldots, \pi_n) = \text{New PageRank Vector} \quad (\text{unknown}) \]

Separate Pages Likely To Be Most Affected

\[ G = \{\text{most affected}\} \quad \overline{G} = \{\text{less affected}\} \quad S = G \cup \overline{G} \]
Updating By Aggregation

Prior Data

\[ Q_{m \times m} = \text{Old Google Matrix} \] (known)

\[ \phi^T = (\phi_1, \phi_2, \ldots, \phi_m) = \text{Old PageRank Vector} \] (known)

Updated Data

\[ P_{n \times n} = \text{New Google Matrix} \] (known)

\[ \pi^T = (\pi_1, \pi_2, \ldots, \pi_n) = \text{New PageRank Vector} \] (unknown)

Separate Pages Likely To Be Most Affected

\[ G = \{ \text{most affected} \} \quad \overline{G} = \{ \text{less affected} \} \quad S = G \cup \overline{G} \]

New pages (and neighbors) go into \( G \)
Aggregation

Partitioned Matrix

$$\mathbf{P}_{n \times n} = \frac{G}{G} \begin{pmatrix} G & \overline{G} \\ P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} = \begin{bmatrix} p_{11} & \cdots & p_{1g} & r^T_1 \\ \vdots & \ddots & \vdots & \vdots \\ p_{g1} & \cdots & p_{gg} & r^T_g \\ c_1 & \cdots & c_g & \mathbf{P}_{22} \end{bmatrix}$$

$$\pi^T = (\pi_1, \ldots, \pi_g \mid \pi_{g+1}, \ldots, \pi_n)$$
Aggregation

Partitioned Matrix

\[
P_{n \times n} = \frac{G}{G} \begin{pmatrix} G & \overline{G} \\ P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} = \begin{bmatrix} p_{11} & \cdots & p_{1g} & r_{1}^T \\ \vdots & \ddots & \vdots & \vdots \\ p_{g1} & \cdots & p_{gg} & r_{g}^T \\ c_{1} & \cdots & c_{g} & P_{22} \end{bmatrix}
\]

\[
\pi^T = (\pi_1, \ldots, \pi_g \mid \pi_{g+1}, \ldots, \pi_n)
\]

Perron Complements

\[p_{11} \cdots p_{gg}\] are \(1 \times 1\) \(\implies\) Perron complements = 1
\[\implies\] Perron vectors = 1
### Aggregation

#### Partitioned Matrix

\[
P_{n \times n} = \frac{G}{G} \begin{pmatrix} G & \overline{G} \\ P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} = \begin{bmatrix} p_{11} & \cdots & p_{1g} & r_1^T \\ \vdots & \ddots & \vdots & \vdots \\ p_{g1} & \cdots & p_{gg} & r_g^T \\ c_1 & \cdots & c_g & P_{22} \end{bmatrix}
\]

\[
\pi^T = (\pi_1, \ldots, \pi_g | \pi_{g+1}, \ldots, \pi_n)
\]

#### Perron Complements

- \(p_{11} \cdots p_{gg}\) are \(1 \times 1\) \implies Perron complements = 1
- \[\implies\] Perron vectors = 1

One significant complement \(S_2 = P_{22} + P_{21}(I - P_{11})^{-1}P_{12}\)

One significant Perron vector \(s_{2}^T S_2 = s_{2}^T\)

A/D Theorem \[\implies\] \(s_{2}^T = (\pi_{g+1}, \ldots, \pi_n) / \sum_{i=g+1}^{n} \pi_i\)
Approximate Aggregation

Use Some Old PageRanks to Approximate New Ones

\((\pi_{g+1}, \ldots, \pi_n) \approx (\phi_{g+1}, \ldots, \phi_n)\)
Approximate Aggregation

Use Some Old PageRanks to Approximate New Ones

\((\pi_{g+1}, \cdots, \pi_n) \approx (\phi_{g+1}, \cdots, \phi_n)\)

Approximate Perron Vector

\[
\mathbf{s}_2^T = \frac{(\pi_{g+1}, \cdots, \pi_n)}{\sum_{i=g+1}^n \pi_i} \approx \frac{(\phi_{g+1}, \cdots, \phi_n)}{\sum_{i=g+1}^n \phi_i} = \tilde{\mathbf{s}}_2^T
\]
Approximate Aggregation

Use Some Old PageRanks to Approximate New Ones

$$(\pi_{g+1}, \cdots, \pi_n) \approx (\phi_{g+1}, \cdots, \phi_n)$$

Approximate Perron Vector

$$s_2^T = \left( \pi_{g+1}, \cdots, \pi_n \right) \approx \left( \phi_{g+1}, \cdots, \phi_n \right) = \tilde{s}_2^T$$

Approximate Aggregation Matrix

$$\tilde{A} = \begin{bmatrix} P_{11} & P_{12}e \\ \tilde{s}_2^T P_{21} & 1 - \tilde{s}_2^T P_{21}e \end{bmatrix} \quad \tilde{\alpha}^T = \left( \tilde{\alpha}_1, \ldots, \tilde{\alpha}_g, \tilde{\alpha}_{g+1} \right)$$
Approximate Aggregation

Use Some Old PageRanks to Approximate New Ones

\[
(\pi_{g+1}, \ldots, \pi_n) \approx (\phi_{g+1}, \ldots, \phi_n)
\]

Approximate Perron Vector

\[
s_2^T = \frac{(\pi_{g+1}, \ldots, \pi_n)}{\sum_{i=g+1}^{n} \pi_i} \approx \frac{(\phi_{g+1}, \ldots, \phi_n)}{\sum_{i=g+1}^{n} \phi_i} = \tilde{s}_2^T
\]

Approximate Aggregation Matrix

\[
\tilde{A} = \begin{bmatrix}
P_{11} & P_{12}e \\
\tilde{s}_2^T P_{21} & 1 - \tilde{s}_2^T P_{21}e
\end{bmatrix}
\]

\[
\tilde{\alpha}^T = (\tilde{\alpha}_1, \ldots, \tilde{\alpha}_g, \tilde{\alpha}_{g+1})
\]

Approximate New PageRank Vector

\[
\tilde{\pi}^T = (\tilde{\alpha}_1, \ldots, \tilde{\alpha}_g | \tilde{\alpha}_{g+1} \tilde{s}_2^T)
\]
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?

NO!

Can’t do A/D twice — a fixed point emerges
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?

NO!

Can’t do A/D twice — a fixed point emerges

Solution

Perturb A/D output to move off of fixed point
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?

NO!
Can’t do A/D twice — a fixed point emerges

Solution

Perturb A/D output to move off of fixed point
Move it in direction of solution
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?

NO!

Can’t do A/D twice — a fixed point emerges

Solution

Perturb A/D output to move off of fixed point
Move it in direction of solution

\[ \tilde{\pi}^T = \tilde{\pi}^T P \]  

(a smoothing step)
Iterative Aggregation

**Improve By Successive Aggregation / Disaggregation?**

NO!

Can’t do A/D twice — a fixed point emerges

**Solution**

Perturb A/D output to move off of fixed point

Move it in direction of solution

$$\tilde{\pi}^T = \tilde{\pi}^T P$$

(a smoothing step)

**The Iterative A/D Updating Algorithm**
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?

NO!
Can’t do A/D twice — a fixed point emerges

Solution

Perturb A/D output to move off of fixed point
Move it in direction of solution

\[ \tilde{\pi}^T = \tilde{\pi}^T P \]  
(a smoothing step)

The Iterative A/D Updating Algorithm

Determine the “G-set” partition \( S = G \cup \overline{G} \)
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?

NO!
Can’t do A/D twice — a fixed point emerges

Solution

Perturb A/D output to move off of fixed point
Move it in direction of solution
\[ \tilde{\pi}^T = \tilde{\pi}^T P \] (a smoothing step)

The Iterative A/D Updating Algorithm

Determine the “G-set” partition \( S = G \cup \overline{G} \)
Approximate A/D step generates approximation \( \tilde{\pi}^T \)
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?

NO!
Can’t do A/D twice — a fixed point emerges

Solution
Perturb A/D output to move off of fixed point
Move it in direction of solution
\[ \tilde{\pi}^T = \tilde{\pi}^T P \]

The Iterative A/D Updating Algorithm
Determine the “G-set” partition \( S = G \cup \overline{G} \)
Approximate A/D step generates approximation \( \tilde{\pi}^T \)
Smooth the result \( \tilde{\pi}^T = \tilde{\pi}^T P \)
Iterative Aggregation

Improve By Successive Aggregation / Disaggregation?

NO!
Can’t do A/D twice — a fixed point emerges

Solution

Perturb A/D output to move off of fixed point
Move it in direction of solution

\[ \tilde{\pi}^T = \tilde{\pi}^T P \]

(a smoothing step)

The Iterative A/D Updating Algorithm

Determine the “G-set” partition \( S = G \cup \overline{G} \)
Approximate A/D step generates approximation \( \tilde{\pi}^T \)
Smooth the result \( \tilde{\tilde{\pi}}^T = \tilde{\pi}^T P \)
Use \( \tilde{\pi}^T \) as input to another approximate aggregation step

\[ \vdots \]
Convergence

THEOREM

✓ Always converges to the new PageRank vector $\pi^T$
THEOREM

✓ Always converges to the new PageRank vector \( \pi^T \)

✓ Converges for all partitions \( S = G \cup \overline{G} \)
Convergence

THEOREM

✓ Always converges to the new PageRank vector $\pi^T$

✓ Converges for all partitions $S = G \cup \bar{G}$

✓ Rate of convergence governed by $|\lambda_2(S_2)|$

$$S_2 = P_{22} + P_{21}(I - P_{11})^{-1}P_{12}$$
Convergence

THEOREM

✓ Always converges to the new PageRank vector \( \pi^T \)
✓ Converges for all partitions \( S = G \cup \overline{G} \)
✓ Rate of convergence governed by \( |\lambda_2(S_2)| \)

\[ S_2 = P_{22} + P_{21}(I - P_{11})^{-1}P_{12} \]

THE GAME

Find a relatively small \( G \) to minimize \( |\lambda_2(S_2)| \)
Convergence

THEOREM

✓ Always converges to the new PageRank vector $\pi^T$

✓ Converges for all partitions $S = G \cup \bar{G}$

✓ Rate of convergence governed by $|\lambda_2(S_2)|$

$S_2 = P_{22} + P_{21}(I - P_{11})^{-1}P_{12}$

THE GAME

Find a relatively small $G$ to minimize $|\lambda_2(S_2)|$

✓ Can do — Use “power law” distribution of the web
Conclusions

Elegant Blend of NA, LA, Graph Theory, MC, & CS
Conclusions

Elegant Blend of NA, LA, Graph Theory, MC, & CS

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

Elegant Blend of NA, LA, Graph Theory, MC, & CS

Google Now Uses Many Other “Metrics” to augment PR

Search Is Opening New Areas Ripe For Inovative Ideas

Exciting Work That Is Changing The World
Conclusions

Elegant Blend of NA, LA, Graph Theory, MC, & CS

Google Now Uses Many Other “Metrics” to augment PR

Search Is Opening New Areas Ripe For Inovative Ideas

Exciting Work That Is Changing The World

Thanks For Your Attention