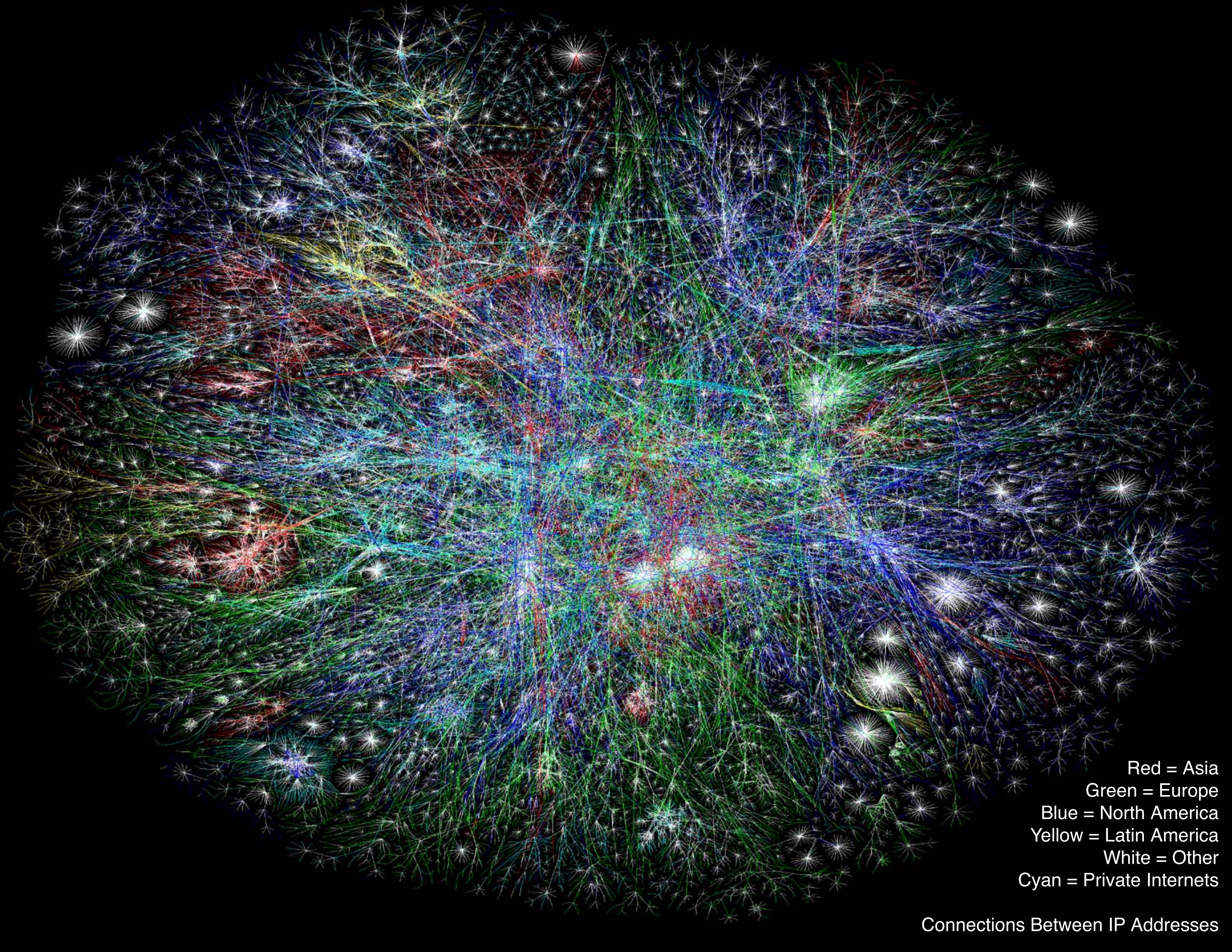
Mathematical Fuel For Search Engines

Carl Meyer
Amy Langville

Department of Mathematics
North Carolina State University
Raleigh, NC

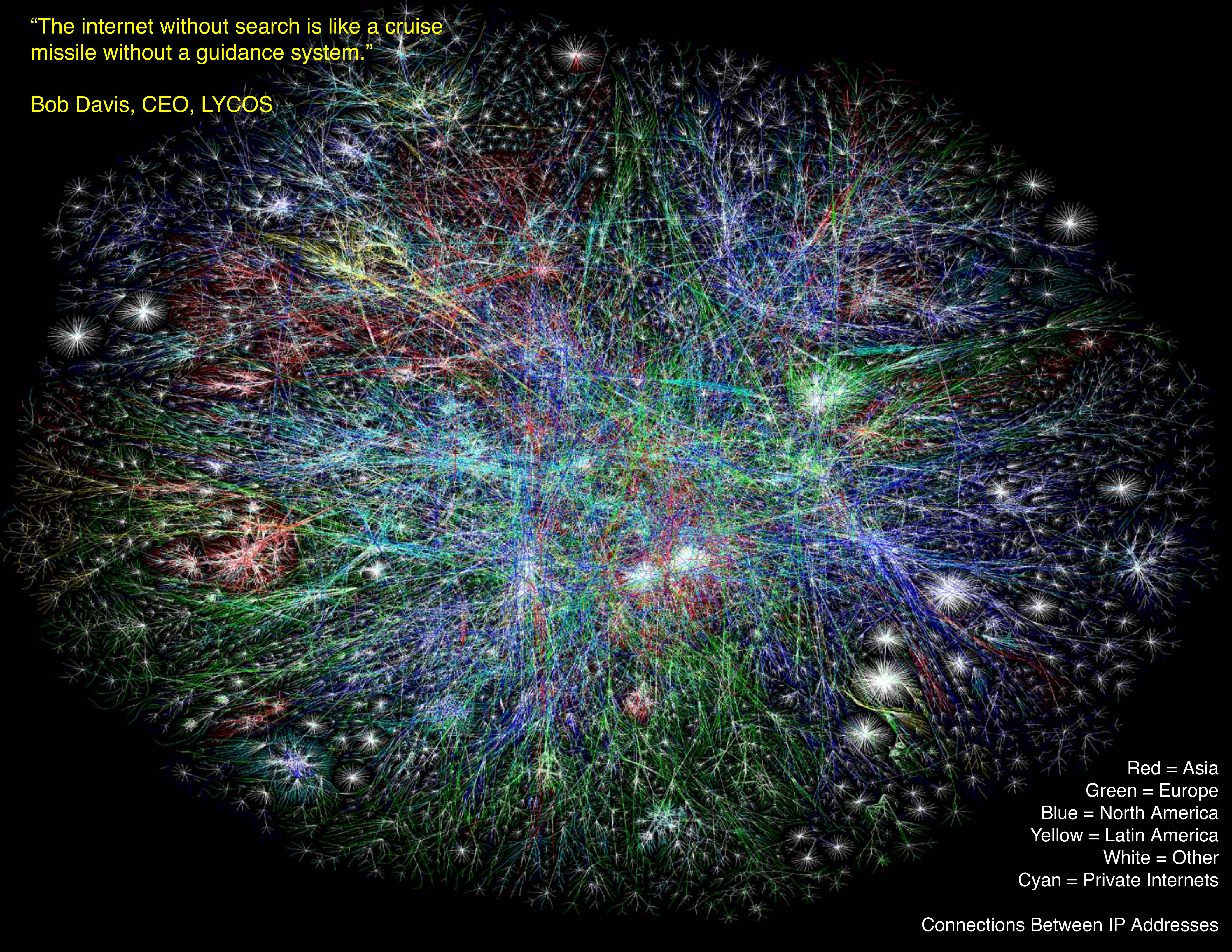
SIAM SEAS 4/2/2004


Red = Asia
Green = Europe
Blue = North America
Yellow = Latin America
White = Other
Cyan = Private Internets

Connections Between IP Addresses

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

Bob Davis, CEO, LYCOS



Red = Asia
Green = Europe
Blue = North America
Yellow = Latin America
White = Other
Cyan = Private Internets

Connections Between IP Addresses

Search Engines

System for the **M**echanical **A**nalysis and **R**etrieval of **T**ext

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} = # times term i appears in document j

Term–Document Matrix

$$\begin{array}{c} \\ \textsc{Term } 1 \\ \textsc{Term } 2 \\ \vdots \\ \textsc{Term } m \end{array} \begin{array}{c} \begin{array}{cccc} \textsc{Doc } 1 & \textsc{Doc } 2 & \cdots & \textsc{Doc } n \end{array} \\ \begin{pmatrix} f_{11} & f_{12} & \cdots & f_{1n} \\ f_{21} & f_{22} & \cdots & f_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ f_{m1} & f_{m2} & \cdots & f_{mn} \end{pmatrix} \end{array} = \mathbf{A}_{m\times n}$$

Query Matching

Query Vector

$$\mathbf{q}^T = (q_1, q_2, \ldots, q_m) \qquad q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$$

Query Matching

Query Vector

$$\mathbf{q}^T = (q_1, q_2, \ldots, q_m) \qquad q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$$

How Close is Query to Each Document?

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

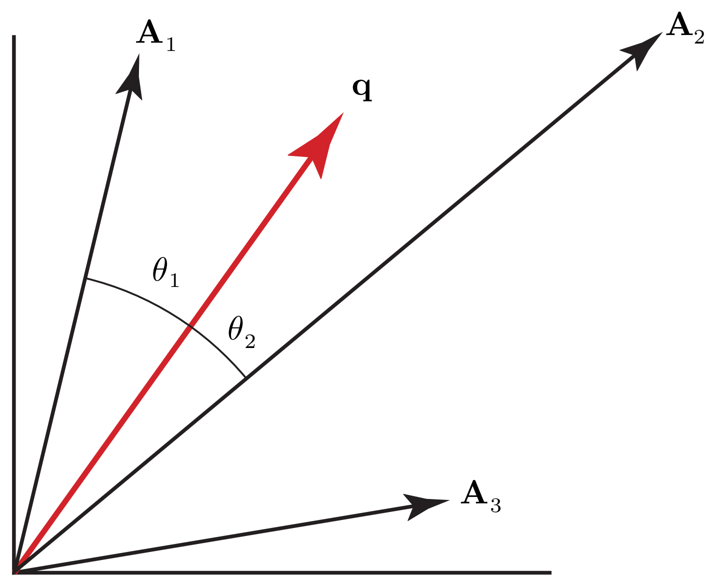
Query Matching

Query Vector

$$\mathbf{q}^T = (q_1, q_2, \ldots, q_m) \qquad q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$$

How Close is Query to Each Document?

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



$$\|\mathbf{q} - \mathbf{A}_1\| < \|\mathbf{q} - \mathbf{A}_2\| \text{ but } \theta_2 < \theta_1$$

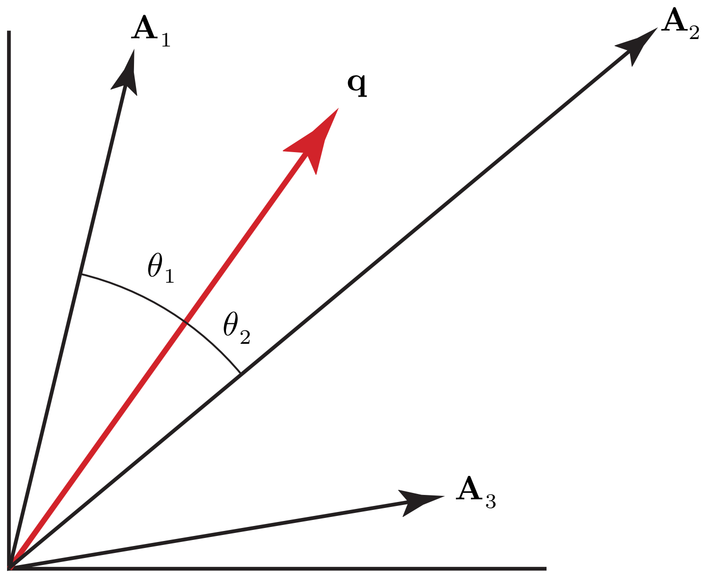
Query Matching

Query Vector

$$\mathbf{q}^T = (q_1, q_2, \ldots, q_m) \qquad q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$$

How Close is Query to Each Document?

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



$$\|\mathbf{q} - \mathbf{A}_1\| < \|\mathbf{q} - \mathbf{A}_2\| \text{ but } \theta_2 < \theta_1$$

$$\text{Use } \delta_i = \cos\theta_i = \frac{\mathbf{q}^T \mathbf{A}_i}{\|\mathbf{q}\| \, \|\mathbf{A}_i\|}$$

Rank documents by size of δ_i

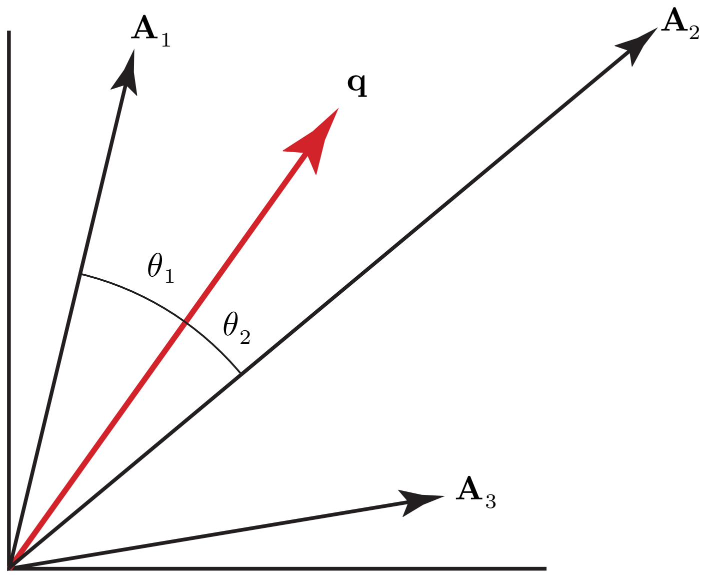
Query Matching

Query Vector

$$\mathbf{q}^T = (q_1, q_2, \ldots, q_m) \qquad q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$$

How Close is Query to Each Document?

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



$$\|\mathbf{q} - \mathbf{A}_1\| < \|\mathbf{q} - \mathbf{A}_2\| \text{ but } \theta_2 < \theta_1$$

$$\text{Use } \delta_i = \cos\theta_i = \frac{\mathbf{q}^T \mathbf{A}_i}{\|\mathbf{q}\|\,\|\mathbf{A}_i\|}$$

Rank documents by size of δ_i

Return Document i to user when $\delta_i \geq tol$

Term Weighting

A Problem

Suppose 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 $\delta_2 \approx 2\delta_1$ (if $\|\mathbf{A}_1\| \approx \|\mathbf{A}_2\|$)

Term Weighting

A Problem

Suppose query = $NCSU$

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

— Then $\delta_2 \approx \mathbf{2}\delta_1$ (if $\|\mathbf{A}_1\| \approx \|\mathbf{A}_2\|$)

To Compensate

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

Term Weighting

A Problem

Suppose query = $NCSU$

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

— Then $\delta_2 \approx \mathbf{2}\delta_1$ (if $\|\mathbf{A}_1\| \approx \|\mathbf{A}_2\|$)

To Compensate

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

Query Weighting

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

Term Weighting

A Problem

Suppose query = $NCSU$

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

— Then $\delta_2 \approx \mathbf{2}\delta_1$ (if $\|\mathbf{A}_1\| \approx \|\mathbf{A}_2\|$)

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 ν_i = # 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 $\|\mathbf{A}_1\| \approx \|\mathbf{A}_2\|$)

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 ν_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 $\cdots$

Uncertainties

Ambiguity in Vocabulary

A $plane$ could be $\cdots$

— A flat geometrical object

Uncertainties

Ambiguity in Vocabulary

A $plane$ could be $\cdots$

- A flat geometrical object
- A woodworking tool

Uncertainties

Ambiguity in Vocabulary

A $plane$ could be $\cdots$

— A flat geometrical object

— A woodworking tool

— A Boeing product

Uncertainties

Ambiguity in Vocabulary

A $plane$ could be $\cdots$

- 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 $\cdots$

- 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 *plane* could be $\cdots$

- 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*

Uncertainties

Ambiguity in Vocabulary

A $plane$ could be $\cdots$

- 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 $\|\mathbf{A}_i\| = 1$

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 $\|\mathbf{q}\| = 1$

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

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

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

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 = *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 $\|\mathbf{A}_i\| = 1$
- 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)
- 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

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i=j, \\ 0 & i \neq j, \end{cases} \qquad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

Latent Semantic Indexing

Use a Fourier expansion of A

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i{=}j, \\ 0 & i{\neq}j, \end{cases} \qquad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = |\, \langle \mathbf{Z}_i | \mathbf{A} \rangle \,| = \text{amount of } \mathbf{A} \text{ in direction of } \mathbf{Z}_i$$

Latent Semantic Indexing

Use a Fourier expansion of A

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i{=}j, \\ 0 & i{\neq}j, \end{cases} \qquad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = |\, \langle \mathbf{Z}_i | \mathbf{A} \rangle \,| = \text{amount of } \mathbf{A} \text{ in direction of } \mathbf{Z}_i$$

Realign data along dominant directions $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_k, \mathbf{Z}_{k+1}, \ldots, \mathbf{Z}_r\}$

— Project **A** onto $span\,\{\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_k\}$

Latent Semantic Indexing

Use a Fourier expansion of A

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i=j, \\ 0 & i\neq j, \end{cases} \qquad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = |\langle \mathbf{Z}_i | \mathbf{A} \rangle| = \text{amount of } \mathbf{A} \text{ in direction of } \mathbf{Z}_i$$

Realign data along dominant directions $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_k, \mathbf{Z}_{k+1}, \ldots, \mathbf{Z}_r\}$

— Project **A** onto $span\,\{\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_k\}$

Truncate: $\mathbf{A}_k = P(\mathbf{A}) = \sigma_1 \mathbf{Z}_1 + \sigma_2 \mathbf{Z}_2 + \cdots + \sigma_k \mathbf{Z}_k$

Latent Semantic Indexing

Use a Fourier expansion of A

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i=j, \\ 0 & i\neq j, \end{cases} \qquad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = | \langle \mathbf{Z}_i | \mathbf{A} \rangle | = \text{amount of } \mathbf{A} \text{ in direction of } \mathbf{Z}_i$$

Realign data along dominant directions $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_k, \mathbf{Z}_{k+1}, \ldots, \mathbf{Z}_r\}$

— Project **A** onto $span\,\{\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_k\}$

Truncate: $\mathbf{A}_k = P(\mathbf{A}) = \sigma_1 \mathbf{Z}_1 + \sigma_2 \mathbf{Z}_2 + \cdots + \sigma_k \mathbf{Z}_k$

LSI: Query matching with $\mathbf{A}_k$ in place of A

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

Latent Semantic Indexing

Use a Fourier expansion of A

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i=j, \\ 0 & i\neq j, \end{cases} \qquad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$|\sigma_i| = |\langle \mathbf{Z}_i | \mathbf{A} \rangle|$ = amount of **A** in direction of $\mathbf{Z}_i$

Realign data along dominant directions $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_k, \mathbf{Z}_{k+1}, \ldots, \mathbf{Z}_r\}$

— Project **A** onto $span\,\{\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_k\}$

Truncate: $\mathbf{A}_k = P(\mathbf{A}) = \sigma_1 \mathbf{Z}_1 + \sigma_2 \mathbf{Z}_2 + \cdots + \sigma_k \mathbf{Z}_k$

LSI: Query matching with $\mathbf{A}_k$ in place of A

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

Possible expansions

— URV: $\mathbf{A} = \mathbf{URV}^T = \sum r_{ij} \mathbf{u}_i \mathbf{v}_j^T$

Latent Semantic Indexing

Use a Fourier expansion of A

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i=j, \\ 0 & i \neq j, \end{cases} \qquad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$|\sigma_i| = |\langle \mathbf{Z}_i | \mathbf{A} \rangle| =$ amount of **A** in direction of $\mathbf{Z}_i$

Realign data along dominant directions $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_k, \mathbf{Z}_{k+1}, \ldots, \mathbf{Z}_r\}$

— Project **A** onto $span\,\{\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_k\}$

Truncate: $\mathbf{A}_k = P(\mathbf{A}) = \sigma_1 \mathbf{Z}_1 + \sigma_2 \mathbf{Z}_2 + \cdots + \sigma_k \mathbf{Z}_k$

LSI: Query matching with $\mathbf{A}_k$ in place of A

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

Possible expansions

— URV: $\mathbf{A} = \mathbf{URV}^T = \sum r_{ij} \mathbf{u}_i \mathbf{v}_j^T$ — SVD: $\mathbf{A} = \mathbf{UDV}^T = \sum \sigma_i \mathbf{u}_i \mathbf{v}_j^T$

Latent Semantic Indexing

Use a Fourier expansion of A

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i=j, \\ 0 & i \neq j, \end{cases} \qquad |\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_r|$$

$$|\sigma_i| = |\langle \mathbf{Z}_i | \mathbf{A} \rangle| = \text{amount of } \mathbf{A} \text{ in direction of } \mathbf{Z}_i$$

Realign data along dominant directions $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_k, \mathbf{Z}_{k+1}, \ldots, \mathbf{Z}_r\}$

— Project **A** onto $span\,\{\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_k\}$

Truncate: $\mathbf{A}_k = P(\mathbf{A}) = \sigma_1 \mathbf{Z}_1 + \sigma_2 \mathbf{Z}_2 + \cdots + \sigma_k \mathbf{Z}_k$

LSI: Query matching with $\mathbf{A}_k$ in place of A

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

Possible expansions

— URV: $\mathbf{A} = \mathbf{URV}^T = \sum r_{ij} \mathbf{u}_i \mathbf{v}_j^T$ — SVD: $\mathbf{A} = \mathbf{UDV}^T = \sum \sigma_i \mathbf{u}_i \mathbf{v}_j^T$

— Haar: $\mathbf{A} = \mathbf{H}_m \mathbf{B} \mathbf{H}_n^T = \sum_{i,j} \beta_{ij} \mathbf{h}_i \mathbf{h}_j^T$ (**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 $\longrightarrow$ to pages)

$$\begin{aligned} Term_{\mathbf{1}} &\rightarrow P_i,\ P_j,\ \ldots \\ Term_{\mathbf{2}} &\rightarrow P_k,\ P_l,\ \ldots \\ &\vdots \end{aligned}$$

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 $\longrightarrow$ to pages)

$$\begin{aligned} Term_1 &\rightarrow P_i,\ P_j,\ \ldots \\ Term_2 &\rightarrow P_k,\ P_l,\ \ldots \\ &\vdots \end{aligned}$$

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 $\longrightarrow$ to pages)

$$Term_{\mathbf{1}} \to P_i,\ P_j,\ \ldots$$
$$Term_{\mathbf{2}} \to P_k,\ P_l,\ \ldots$$
$$\vdots$$

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 $\longrightarrow$ 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: $P_i \sim r_i$
  - r_i based only on link structure (i.e., query independent)
  - 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 $\longrightarrow$ to pages)

$$\begin{aligned} Term_1 &\rightarrow P_i,\; P_j,\; \ldots \\ Term_2 &\rightarrow P_k,\; P_l,\; \ldots \\ &\vdots \end{aligned}$$

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

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

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 $\longrightarrow$ to pages)

$$\begin{aligned} Term_1 &\rightarrow P_i,\ P_j,\ \ldots \\ Term_2 &\rightarrow P_k,\ P_l,\ \ldots \\ &\vdots \end{aligned}$$

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

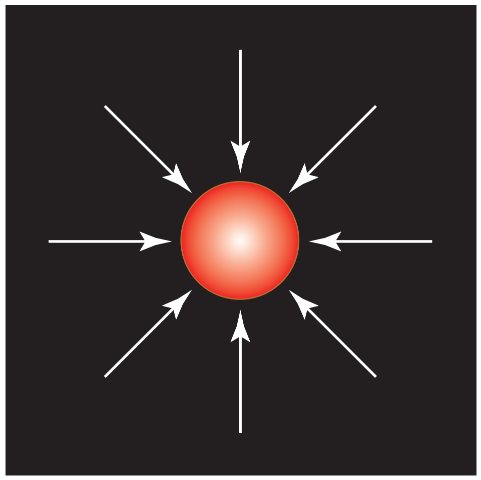
Direct Query Matching

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

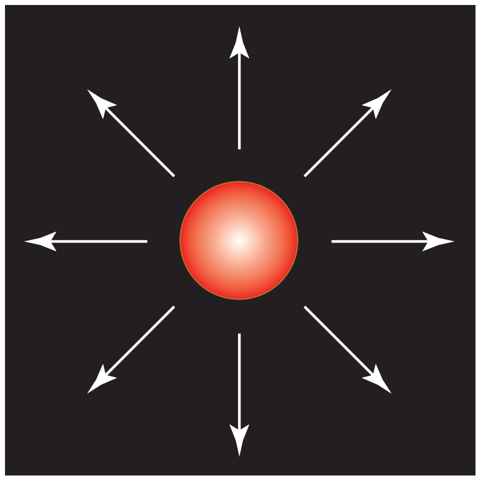
Return $P_i, P_j, P_k,\ \ldots$ in order of ranks $r_i, r_j, r_k, \ldots$

How To Measure “Importance”

Authorities

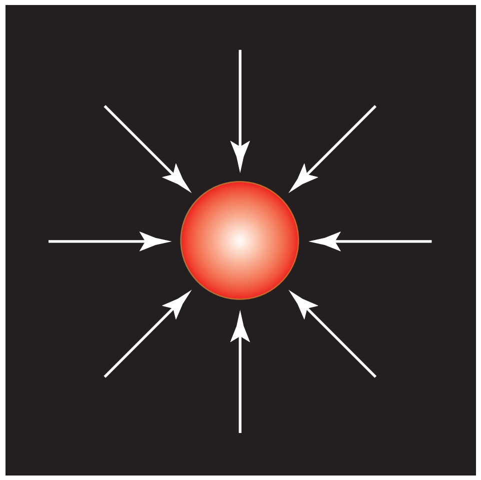


Hubs

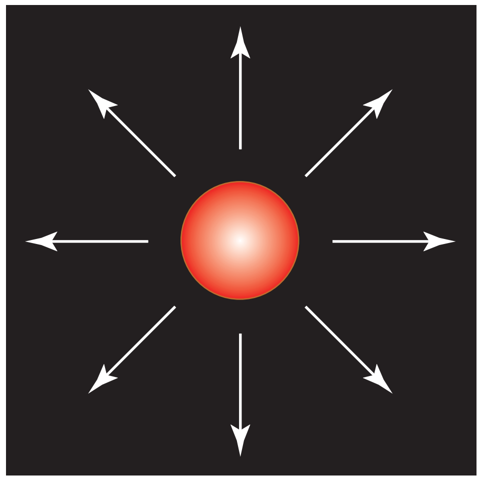


How To Measure “Importance”

Authorities



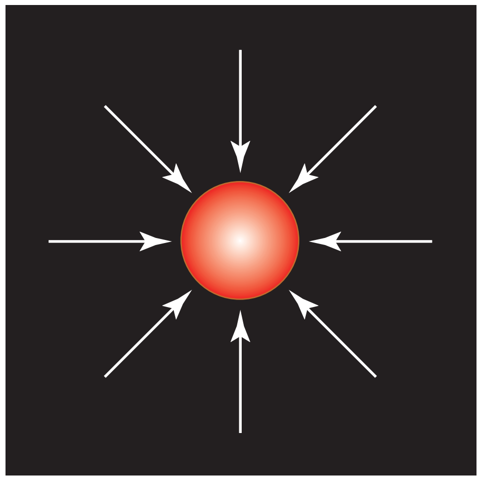
Hubs



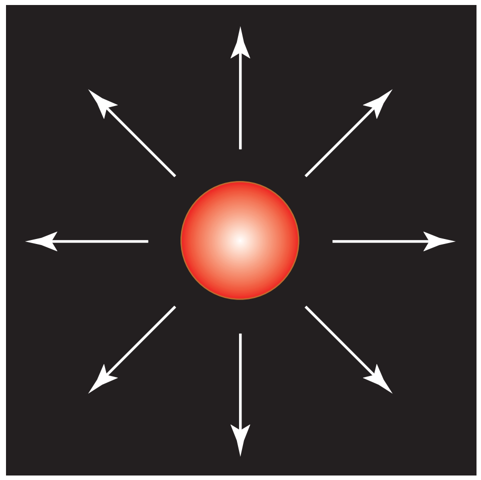
- Good hub pages point to good authority pages

How To Measure "Importance"

Authorities



Hubs



- 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 = authority score for P_i
- h_i = hub score for 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

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 = 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

$$\mathbf{L}_{ij} = \begin{cases} 1 & P_i \to P_j \\ 0 & P_i \not\to P_j \end{cases}$$

- Successively refine rankings
  - For $k = 1, 2, \ldots$

$$a_i(k) = \sum_{j:P_j \to P_i} h_j(k-1) \quad \Rightarrow \quad \mathbf{a}_k = \mathbf{L}^T \mathbf{h}_{k-1}$$

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

$$\mathbf{L}_{ij} = \begin{cases} 1 & P_i \rightarrow P_j \\ 0 & P_i \not\rightarrow P_j \end{cases}$$

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

$$a_i(k) = \sum_{j:P_j \rightarrow P_i} h_j(k-1) \quad \Rightarrow \quad \mathbf{a}_k = \mathbf{L}^T \mathbf{h}_{k-1}$$

$$h_i(k) = \sum_{j:P_i \rightarrow P_j} a_j(k)$$

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

$$\mathbf{L}_{ij} = \begin{cases} 1 & P_i \rightarrow P_j \\ 0 & P_i \not\rightarrow P_j \end{cases}$$

- Successively refine rankings
  - For $k = 1, 2, \ldots$

$$a_i(k) = \sum_{j:P_j \rightarrow P_i} h_j(k-1) \quad \Rightarrow \quad \mathbf{a}_k = \mathbf{L}^T \mathbf{h}_{k-1}$$

$$h_i(k) = \sum_{j:P_i \rightarrow P_j} a_j(k) \quad \Rightarrow \quad \mathbf{h}_k = \mathbf{L}\mathbf{a}_k$$

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

$$\mathbf{L}_{ij} = \begin{cases} 1 & P_i \rightarrow P_j \\ 0 & P_i \not\rightarrow P_j \end{cases}$$

- Successively refine rankings
  - For $k = 1, 2, \ldots$

$$a_i(k) = \sum_{j:P_j \rightarrow P_i} h_j(k-1) \quad \Rightarrow \quad \mathbf{a}_k = \mathbf{L}^T \mathbf{h}_{k-1}$$

$$h_i(k) = \sum_{j:P_i \rightarrow P_j} a_j(k) \quad \Rightarrow \quad \mathbf{h}_k = \mathbf{L}\mathbf{a}_k$$

  - $\mathbf{A} = \mathbf{L}^T\mathbf{L}$ $\quad \mathbf{a}_k = \mathbf{A}\mathbf{a}_{k-1} \rightarrow$ 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

$$\mathbf{L}_{ij} = \begin{cases} 1 & P_i \rightarrow P_j \\ 0 & P_i \not\rightarrow P_j \end{cases}$$

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

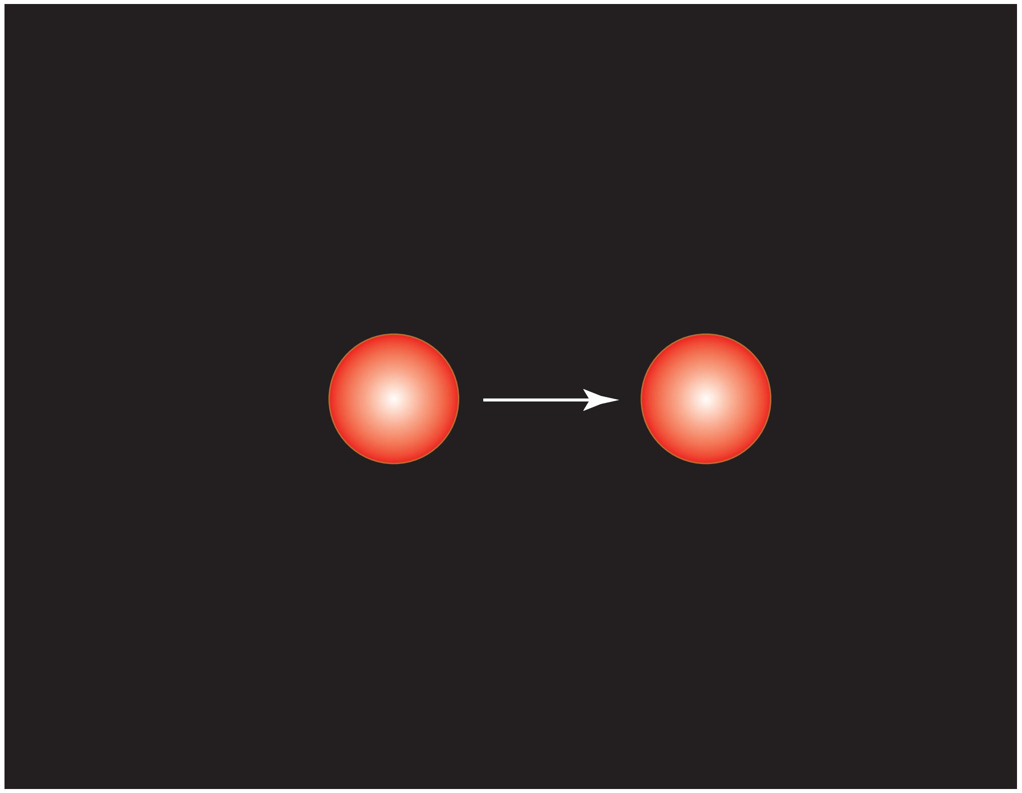
$$a_i(k) = \sum_{j:P_j \rightarrow P_i} h_j(k-1) \quad \Rightarrow \quad \mathbf{a}_k = \mathbf{L}^T \mathbf{h}_{k-1}$$

$$h_i(k) = \sum_{j:P_i \rightarrow P_j} a_j(k) \quad \Rightarrow \quad \mathbf{h}_k = \mathbf{L}\mathbf{a}_k$$

  - $\mathbf{A} = \mathbf{L}^T\mathbf{L}$ $\quad \mathbf{a}_k = \mathbf{A}\mathbf{a}_{k-1} \rightarrow$ e-vector (direction)
  - $\mathbf{H} = \mathbf{L}\mathbf{L}^T$ $\quad \mathbf{h}_k = \mathbf{H}\mathbf{h}_{k-1} \rightarrow$ e-vector (direction)

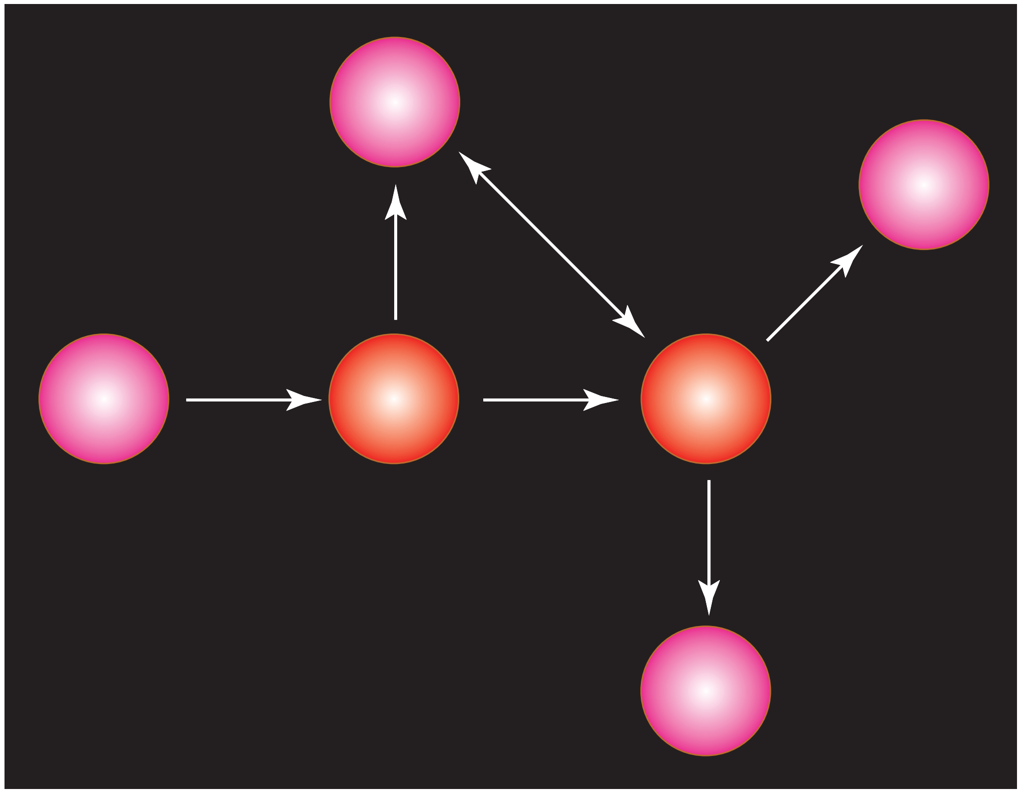
Compromise

1. Do direct query matching



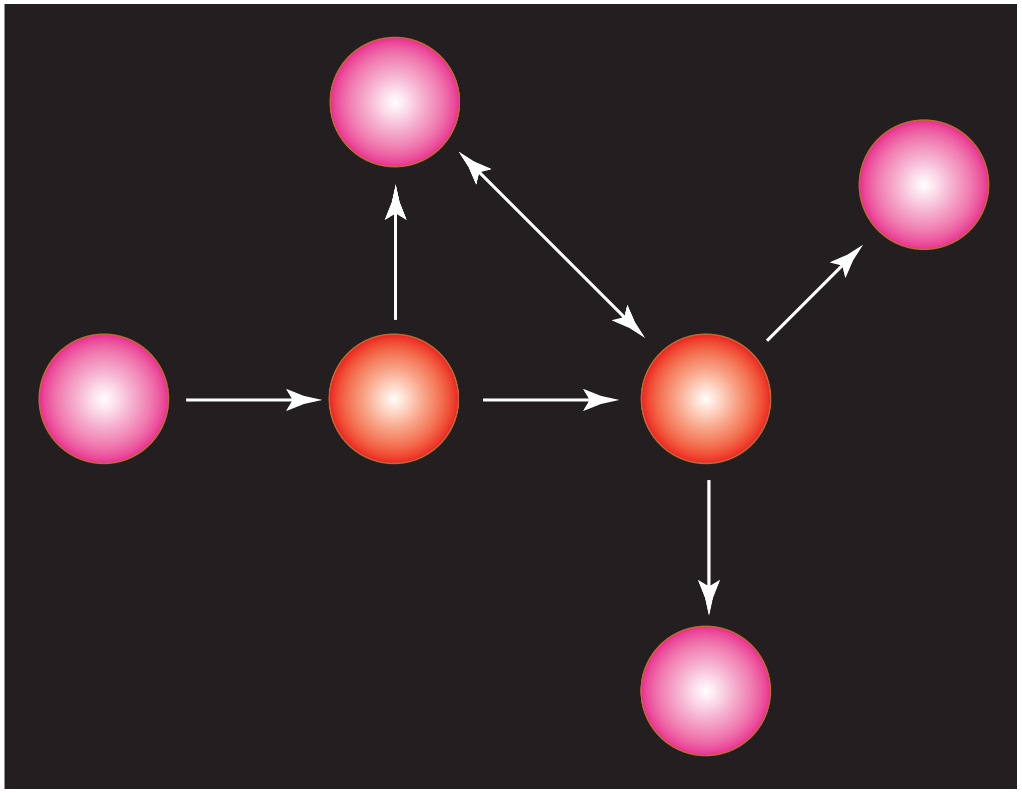
Compromise

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

Newsweek

March 29, 2004 : $3.95

newsweek.msnbc.com

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)

PageRank $r(P)$ Is Not Query Dependent

Google's PageRank

(Lawrence Page & Sergey Brin 1998)

PageRank $r(P)$ Is Not Query Dependent

- Depends primarily on link structure of web

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

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

(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

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" $\Longrightarrow$ 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" $\Longrightarrow$ 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

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

$\mathcal{B}_P$ = $\{$all pages pointing to $P\}$

$|P|$ = number of out links from P

PageRank

The Definition

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

$\mathcal{B}_P$ = {all pages pointing to P}

$|P|$ = 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

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

$\mathcal{B}_P$ = {all pages pointing to P}

$|P|$ = 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

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

$\mathcal{B}_P$ = $\{$all pages pointing to $P\}$

$|P|$ = 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

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

PageRank

The Definition

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

$\mathcal{B}_P$ = {all pages pointing to P}

$|P|$ = 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

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

$\mathcal{B}_P$ = {all pages pointing to P}

$|P|$ = 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

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

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

$$\ddots$$

$$r_{j+1}(P_i) = \sum_{P \in \mathcal{B}_{P_i}} \frac{r_j(P)}{|P|}$$

In Matrix Notation

After Step j

$$\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n) \right]$$

In Matrix Notation

After Step j

$$\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n)\right]$$

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

In Matrix Notation

After Step j

$$\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n) \right]$$

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

$$\text{PageRank} = \lim_{j\to\infty} \boldsymbol{\pi}_j^T = \boldsymbol{\pi}^T$$

(provided limit exists)

In Matrix Notation

After Step j

$$\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n) \right]$$

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

$$\text{PageRank} = \lim_{j \rightarrow \infty} \boldsymbol{\pi}_j^T = \boldsymbol{\pi}^T$$

(provided limit exists)

It’s A Markov Chain

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

(set p_{ii}=1 when all other p_{ij}=0)

In Matrix Notation

After Step j

$$\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n)\right]$$

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

$$\text{PageRank} = \lim_{j\rightarrow\infty} \boldsymbol{\pi}_j^T = \boldsymbol{\pi}^T$$

(provided limit exists)

It's A Markov Chain

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

Each $\boldsymbol{\pi}_j^T$ is a probability distribution vector $\left(\sum_i r_j(P_i)=1\right)$

In Matrix Notation

After Step j

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

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

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

It's A Markov Chain

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

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

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

In Matrix Notation

After Step j

$$\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n)\right]$$

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

$$\text{PageRank} = \lim_{j\to\infty} \boldsymbol{\pi}_j^T = \boldsymbol{\pi}^T$$

(provided limit exists)

It's A Markov Chain

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

Each $\boldsymbol{\pi}_j^T$ is a probability distribution vector $\left(\sum_i r_j(P_i)=1\right)$

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

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

Random Surfer

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

Long-run proportion of time on page P_i is π_i

Random Surfer

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

Long-run proportion of time on page P_i is π_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 π_i

Problems

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

✓ π^T not well defined

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

Long-run proportion of time on page P_i is π_i

Problems

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

✓ π^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

(Back button not a link)

Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

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

✓ $\boldsymbol{\pi}^T$ 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 π_i

Problems

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

- ✓ π^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

(Back button not a link)

Long-run proportion of time on page P_i is π_i

Problems

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

✓ π^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

(Back button not a link)

Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

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

✓ $\boldsymbol{\pi}^T$ not well defined

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

✓ No convergence

Convergence

Markov chain must be irreducible and aperiodic

Bored Surfer Enters Random URL

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

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

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

- ✓ $\boldsymbol{\pi}^T$ 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 $\mathbf{P}$ by $\widetilde{\mathbf{P}} = \alpha\mathbf{P} + (1-\alpha)\mathbf{E}$ $\quad e_{ij} = 1/n \quad \alpha \approx .85$

Different $\mathbf{E} = \mathbf{e}\mathbf{v}^T$ and α allow customization & speedup

THE WALL STREET JOURNAL.

© 2003 Dow Jones & Company. All Rights Reserved

WEDNESDAY, FEBRUARY 26, 2003 - VOL. CCXLI NO. 39 - ★★★ $1.00 •• WSJ.com

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)

* * *

■ **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.

(Articles on Page A2)

* * *

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

(Article on Page A3)

* * *

■ **The industrials** 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 imprimatur. 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 build-up. (Column 4 and Pages A4 and A6)

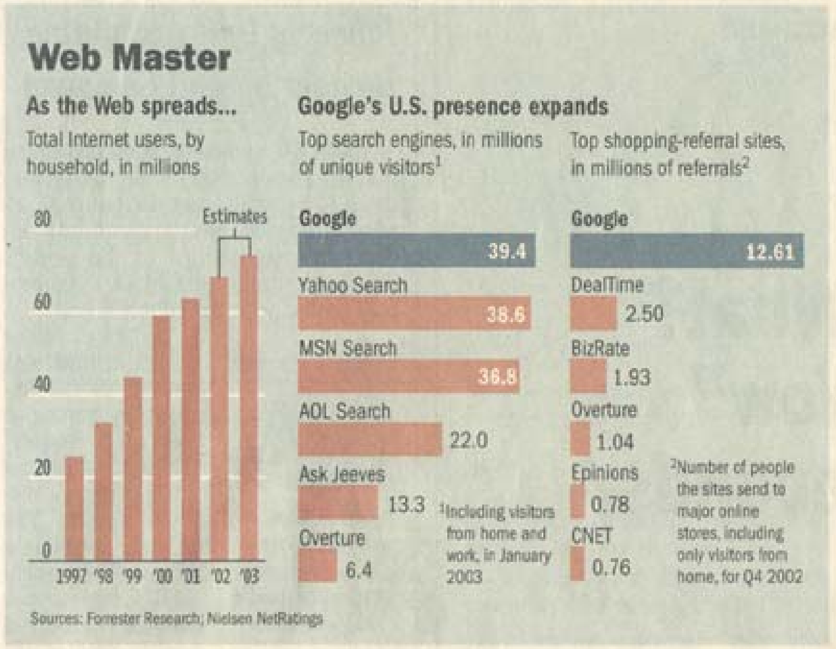
Turkey's parliament debated legislation to let the U.S. deploy 62,000 to open a northern front. Kurdish soldiers lined roads in a show of force as U.S. officials traveled into Iraq's north for an opposition conference.

* * *

■ **Powell said** North Korea hasn't restarted a reactor and plutonium-processing facility at Yongbyon, hinting such 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 A14)

* * *

■ **The FBI came under** withering bipartisan criticism in a Senate Judi-



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.

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.

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 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 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 coding, called "metatags." The coding fools a search engine 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 ranking a site. But Google's system, based on links, wasn't fooled.

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 clogged 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 Google bombing. Pranksters would try to

creating Web sites that were nothing more than collections of links to the clients' site, 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

'The big search engines determine the laws of how commerce runs,' says Mr. Massa.

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 Mesa, Ariz., she quickly learned the importance of appearing near the top of search-engine results, especially on Google. She boned 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 help steer a search toward her site when a user entered words such as "haltertops" 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 site's search-engine rankings by taking

In theory, when Google encounters the AutomatedLinks code, it treats it as a legitimate referral to the other sites and counts them in toting up the sites' popularity.

Shortly after Ms. Holman signed up with AutomatedLinks in July, she read on an online discussion group that Google objected to such link arrangements. 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 still remained on the sites of other AutomatedLinks customers. Then, sometime in November, her site was suddenly no longer appearing among the top results. Her orders plunged as much as 80%.

Ms. Holman, who e-mailed Google and AutomatedLinks, says she has been unable to get answers. But in the last few months, other AutomatedLinks customers say they have seen their sites apparently penalized by Google. Graham McLeay, who runs 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 Website 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 declined to comment on the case. But 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 that using 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 visitors to a single site.

To stay one step ahead of the Web

homa City-based SearchKing, an online directory for hundreds of small, specialty Web sites. SearchKing also sells advertising links designed both to deliver traffic to an advertiser and boost its rankings in Google and other search results.

Bob Massa, SearchKing's chief executive, last August launched the PR Ad Network as a way to capitalize on Google's page-ranking system, known as PageRank. PageRank rates Web sites on a scale of one to 10 based on their popularity, and the rankings can be viewed by Web users if they install special Google software. PR Ad Network sells ads that are priced according to a site's PageRank, with higher-ranked sites commanding higher prices. When a site buys an advertising link on a highly ranked site, the ad buyer could see its ratings improve because of the greater weight Google gives to that link.

Shortly after publicizing the ad network, Mr. Massa discovered that his site suddenly dropped in Google's rankings. What's more, sites that participated in the separate SearchKing directory also had their Google rankings lowered. He filed a lawsuit in Oklahoma City federal court, claiming Google was punishing him for trying to profit from the company's page-ranking system.

A Google spokesman won't comment on the case. In its court filings, Google said it demoted pages on the SearchKing site because of SearchKing's attempts to manipulate search results. The company has asked for the suit to be dismissed, arguing that the PageRank represents its opinion of the value of a Web site and as such is protected by the First Amendment.

"The big search engines determine the laws of how commerce runs," says Mr. Massa, who is persisting with the lawsuit even though the sites have had their page rankings partly restored. "Someone needs to demand accountability."

Google is taking steps that many say could satisfy businesses trying to boost their rankings. Google has long sold sponsored links that show up on the top of many search-results pages, separate from the main listings. Last year, the company expanded its paid-listings program, so that there are now more slots where sites can pay for a prominent place in the results. Many sites now are turning to advertising instead of tactics to optimize their rankings.

Home Depo
Amid First

By Chad Terhun

ATLANTA—**Home Depot I**
fiscal fourth-quarter earnin
3.4% on disappointing sales.
Speaking to investors a
analysts, the company's ch
chief executive, Bob Na
Home Depot is prepared t
dissatisfied customers and
competitive challenge from
val with remodeled stores, i
ventory and improved custo
The nation's largest ho
ment retailer said net income
ter ended Feb. 2 decreased to
or 30 cents a share, from $71
30 cents a share, a year earli
2% to $13.21 billion from $13.4
first quarterly sales decline i
ny's 24-year history. Home
the latest quarter was a week
a year earlier. Using compara
periods, the company said qu
increased 5% and net income
Same-store sales, or sal
open at least a year, decline
quarter. Home Depot said st
last month offset a disastrou
and helped the retailer avoi
estimate that same-store sale
as much as 10%. In 4 p.m
Stock Exchange composite tr
Depot shares rose 66 cents t

Fiat Patria
Is Set to Bec

By Alessandra Ga

ROME—Umberto Agnelli
named **Fiat SpA** chairman on
ping into the driver's seat as th
glomerate works on an 11th-h
ing of its unprofitable car un
Mr. Agnelli, the 68-year-o
Fiat patriarch Gianni Agne
last month, was widely expe
over from current chair
Fresco, later this year. But
who has served as chairma

Computing π^T

A Big Problem

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

Computing π^T

A Big Problem

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

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

THE WORLD'S LARGEST MATRIX COMPUTATION

Google's PageRank is an eigenvector of a matrix of order 2.7 billion.

BY CLEVE MOLER

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

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 π^T

A Big Problem

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

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

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

Computing π^T

A Big Problem

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

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

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

Convergence Time

Measured in days

Computing π^T

A Big Problem

Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T\mathbf{P}$ (eigenvector problem)

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

Start with $\boldsymbol{\pi}_0^T = \mathbf{e}/n$ and iterate $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T\mathbf{P}$ (power method)

Convergence Time

Measured in days

A Bigger Problem — Updating

Pages & links are added, deleted, changed continuously

Computing π^T

A Big Problem

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

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

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

A Big Problem

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

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

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

Perron Complementation

Perron Frobenius

$$\mathbf{P} \geq 0, \text{ irreducible} \quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)}$$

Perron Complementation

Perron Frobenius

$$\mathbf{P} \geq 0, \text{ irreducible} \quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)}$$

Unique Left-Hand Perron Vector

$$\boldsymbol{\pi}^T\mathbf{P} = \rho\boldsymbol{\pi}^T \qquad \boldsymbol{\pi}^T > 0 \qquad \|\boldsymbol{\pi}^T\|_1 = 1$$

Perron Complementation

Perron Frobenius

$$\mathbf{P} \geq 0, \text{ irreducible} \quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)}$$

Unique Left-Hand Perron Vector

$$\pi^T\mathbf{P} = \rho\pi^T \qquad \pi^T > 0 \qquad \|\pi^T\|_1 = 1$$

Partition & Aggregate

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

Perron Complementation

Perron Frobenius

$$\mathbf{P} \geq 0, \text{ irreducible} \quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)}$$

Unique Left-Hand Perron Vector

$$\boldsymbol{\pi}^T\mathbf{P} = \rho\boldsymbol{\pi}^T \qquad \boldsymbol{\pi}^T > 0 \qquad \|\boldsymbol{\pi}^T\|_1 = 1$$

Partition & Aggregate

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

Shift **P** by ρ

Perron Complementation

Perron Frobenius

$$\mathbf{P} \geq 0, \text{ irreducible} \quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)}$$

Unique Left-Hand Perron Vector

$$\boldsymbol{\pi}^T\mathbf{P} = \rho\boldsymbol{\pi}^T \qquad \boldsymbol{\pi}^T > 0 \qquad \|\boldsymbol{\pi}^T\|_1 = 1$$

Partition & Aggregate

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

Shift **P** by ρ $\longrightarrow$ Schur Complements

Perron Complementation

Perron Frobenius

$$\mathbf{P} \geq 0, \text{ irreducible} \quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)}$$

Unique Left-Hand Perron Vector

$$\pi^T\mathbf{P} = \rho\pi^T \qquad \pi^T > 0 \qquad \|\pi^T\|_1 = 1$$

Partition & Aggregate

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

Shift $\mathbf{P}$ by ρ $\longrightarrow$ Schur Complements $\longrightarrow$ Shift back by ρ

Perron Complementation

Perron Frobenius

$$\mathbf{P} \geq 0, \text{ irreducible} \quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)}$$

Unique Left-Hand Perron Vector

$$\pi^T\mathbf{P} = \rho\pi^T \qquad \pi^T > 0 \qquad \|\pi^T\|_1 = 1$$

Partition & Aggregate

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

Shift $\mathbf{P}$ by ρ $\longrightarrow$ Schur Complements $\longrightarrow$ Shift back by ρ

Perron Complements

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

Perron Complementation

Perron Frobenius

$\mathbf{P} \geq 0$, irreducible $\quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad$ (simple)

Unique Left-Hand Perron Vector

$\pi^T\mathbf{P} = \rho\pi^T \qquad \pi^T > 0 \qquad \|\pi^T\|_1 = 1$

Partition & Aggregate $\quad \mathbf{P} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{bmatrix}$

Shift $\mathbf{P}$ by ρ $\longrightarrow$ Schur Complements $\longrightarrow$ Shift back by ρ

Perron Complements

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

Inherited Properties

$\mathbf{S}_i \geq 0$

Perron Complementation

Perron Frobenius

$\mathbf{P} \geq 0$, irreducible $\quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad$ (simple)

Unique Left-Hand Perron Vector

$\boldsymbol{\pi}^T\mathbf{P} = \rho\boldsymbol{\pi}^T \qquad \boldsymbol{\pi}^T > 0 \qquad \|\boldsymbol{\pi}^T\|_1 = 1$

Partition & Aggregate

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

Shift $\mathbf{P}$ by ρ $\longrightarrow$ Schur Complements $\longrightarrow$ Shift back by ρ

Perron Complements

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

Inherited Properties

$\mathbf{S}_i \geq 0$

$\mathbf{S}_i$ is irreducible

Perron Complementation

Perron Frobenius

$$\mathbf{P} \geq 0, \text{ irreducible} \quad \Longrightarrow \quad \rho(\mathbf{P}) = \rho \in \sigma(\mathbf{P}) \quad \text{(simple)}$$

Unique Left-Hand Perron Vector

$$\boldsymbol{\pi}^T\mathbf{P} = \rho\boldsymbol{\pi}^T \qquad \boldsymbol{\pi}^T > 0 \qquad \|\boldsymbol{\pi}^T\|_1 = 1$$

Partition & Aggregate

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

Shift $\mathbf{P}$ by ρ $\longrightarrow$ Schur Complements $\longrightarrow$ Shift back by ρ

Perron Complements

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

Inherited Properties

$\mathbf{S}_i \geq 0$

$\mathbf{S}_i$ is irreducible

$\rho(\mathbf{S}_i) = \rho = \rho(\mathbf{P})$

Exact Aggregation

Aggregation Matrix

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

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

Exact Aggregation

Aggregation Matrix

$\mathbf{s}_i^T$ = 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

$$\mathbf{A} \geq 0$$

Exact Aggregation

Aggregation Matrix

$\mathbf{s}_i^T$ = 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

$\mathbf{A} \geq 0$

$\mathbf{A}$ is irreducible

Exact Aggregation

Aggregation Matrix

$\mathbf{s}_i^T$ = 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

$\mathbf{A} \geq 0$

$\mathbf{A}$ is irreducible

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

Exact Aggregation

Aggregation Matrix

$\mathbf{s}_i^T$ = 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

$\mathbf{A} \geq 0$

$\mathbf{A}$ is irreducible

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

The Aggregation/Disaggregation Theorem

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

$\Longrightarrow$

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

$\mathbf{Q}_{m\times m}$ = Old Google Matrix (known)

$\phi^T = (\phi_1, \phi_2, \ldots, \phi_m)$ = Old PageRank Vector (known)

Updating By Aggregation

Prior Data

$\mathbf{Q}_{m\times m}$ = Old Google Matrix (known)

$\boldsymbol{\phi}^T = (\phi_1,\ \phi_2,\ \ldots,\ \phi_m)$ = Old PageRank Vector (known)

Updated Data

$\mathbf{P}_{n\times n}$ = New Google Matrix (known)

$\boldsymbol{\pi}^T = (\pi_1,\ \pi_2,\ \ldots,\ \pi_n)$ = New PageRank Vector (unknown)

Updating By Aggregation

Prior Data

$\mathbf{Q}_{m\times m}$ = Old Google Matrix (known)

$\phi^T = (\phi_1, \phi_2, \ldots, \phi_m)$ = Old PageRank Vector (known)

Updated Data

$\mathbf{P}_{n\times n}$ = New Google Matrix (known)

$\boldsymbol{\pi}^T = (\pi_1, \pi_2, \ldots, \pi_n)$ = New PageRank Vector (unknown)

Separate Pages Likely To Be Most Affected

$G = \{\text{most affected}\} \qquad \overline{G} = \{\text{less affected}\} \qquad \mathcal{S} = G \cup \overline{G}$

Updating By Aggregation

Prior Data

$\mathbf{Q}_{m\times m}$ = Old Google Matrix (known)

$\boldsymbol{\phi}^T = (\phi_1,\ \phi_2,\ \ldots,\ \phi_m)$ = Old PageRank Vector (known)

Updated Data

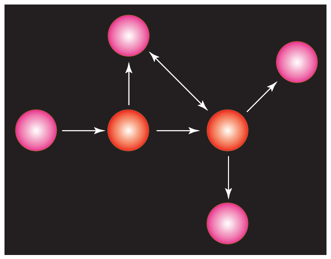
$\mathbf{P}_{n\times n}$ = New Google Matrix (known)

$\boldsymbol{\pi}^T = (\pi_1,\ \pi_2,\ \ldots,\ \pi_n)$ = New PageRank Vector (unknown)

Separate Pages Likely To Be Most Affected

$G = \{\text{most affected}\}$ $\quad \overline{G} = \{\text{less affected}\}$ $\quad \mathcal{S} = G \cup \overline{G}$

New pages (and neighbors) go into G



Aggregation

Partitioned Matrix

$$\mathbf{P}_{n\times n} = \begin{array}{c|cc} & G & \overline{G} \\ \hline G & \mathbf{P}_{11} & \mathbf{P}_{12} \\ \overline{G} & \mathbf{P}_{21} & \mathbf{P}_{22} \end{array} = \left[\begin{array}{c|c|c|c} p_{11} & \cdots & p_{1g} & \mathbf{r}_1^T \\ \hline \vdots & \ddots & \vdots & \vdots \\ \hline p_{g1} & \cdots & p_{gg} & \mathbf{r}_g^T \\ \hline \mathbf{c}_1 & \cdots & \mathbf{c}_g & \mathbf{P}_{22} \end{array}\right]$$

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

Aggregation

Partitioned Matrix

$$\mathbf{P}_{n\times n} = \begin{array}{c c} & \begin{array}{cc} G & \overline{G} \end{array} \\ \begin{array}{c} G \\ \overline{G} \end{array} & \begin{pmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{pmatrix} \end{array} = \left[\begin{array}{c|c|c|c} p_{11} & \cdots & p_{1g} & \mathbf{r}_1^T \\ \hline \vdots & \ddots & \vdots & \vdots \\ \hline p_{g1} & \cdots & p_{gg} & \mathbf{r}_g^T \\ \hline \mathbf{c}_1 & \cdots & \mathbf{c}_g & \mathbf{P}_{22} \end{array}\right]$$

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

Perron Complements

$p_{11} \cdots p_{gg}$ are 1×1 $\Longrightarrow$ Perron complements $= 1$

$\Longrightarrow$ Perron vectors $= 1$

Aggregation

Partitioned Matrix

$$\mathbf{P}_{n\times n} = \begin{array}{c|cc} & G & \overline{G} \\ \hline G & \mathbf{P}_{11} & \mathbf{P}_{12} \\ \overline{G} & \mathbf{P}_{21} & \mathbf{P}_{22} \end{array} = \left[\begin{array}{c|c|c|c} p_{11} & \cdots & p_{1g} & \mathbf{r}_1^T \\ \hline \vdots & \ddots & \vdots & \vdots \\ \hline p_{g1} & \cdots & p_{gg} & \mathbf{r}_g^T \\ \hline \mathbf{c}_1 & \cdots & \mathbf{c}_g & \mathbf{P}_{22} \end{array}\right]$$

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

Perron Complements

$p_{11} \cdots p_{gg}$ are 1×1 $\Longrightarrow$ Perron complements $= 1$

$\Longrightarrow$ Perron vectors $= 1$

One significant complement $\mathbf{S}_2 = \mathbf{P}_{22} + \mathbf{P}_{21}(\mathbf{I} - \mathbf{P}_{11})^{-1}\mathbf{P}_{12}$

One significant Perron vector $\mathbf{s}_2^T\mathbf{S}_2 = \mathbf{s}_2^T$

A/D Theorem $\Longrightarrow$ $\mathbf{s}_2^T = (\pi_{g+1}, \ldots, \pi_n) / \sum_{i=g+1}^{n} \pi_i$

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

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

Approximate Perron Vector

$$\mathbf{s}_2^T = \frac{(\pi_{g+1}, \ldots, \pi_n)}{\sum_{i=g+1}^{n} \pi_i} \approx \frac{(\phi_{g+1}, \ldots, \phi_n)}{\sum_{i=g+1}^{n} \phi_i} = \widetilde{\mathbf{s}}_2^T$$

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

$$\mathbf{s}_2^T=\frac{(\pi_{g+1},\ldots,\pi_n)}{\sum_{i=g+1}^{n}\pi_i}\approx\frac{(\phi_{g+1},\ldots,\phi_n)}{\sum_{i=g+1}^{n}\phi_i}=\widetilde{\mathbf{s}}_2^T$$

Approximate Aggregation Matrix

$$\widetilde{\mathbf{A}}=\begin{bmatrix}\mathbf{P}_{11} & \mathbf{P}_{12}\mathbf{e}\\ \widetilde{\mathbf{s}}_2^T\mathbf{P}_{21} & 1-\widetilde{\mathbf{s}}_2^T\mathbf{P}_{21}\mathbf{e}\end{bmatrix}$$

$$\widetilde{\boldsymbol{\alpha}}^T=\left(\widetilde{\alpha}_1,\ldots,\widetilde{\alpha}_g,\widetilde{\alpha}_{g+1}\right)$$

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

$$\mathbf{s}_2^T = \frac{(\pi_{g+1}, \ldots, \pi_n)}{\sum_{i=g+1}^{n} \pi_i} \approx \frac{(\phi_{g+1}, \ldots, \phi_n)}{\sum_{i=g+1}^{n} \phi_i} = \widetilde{\mathbf{s}}_2^T$$

Approximate Aggregation Matrix

$$\widetilde{\mathbf{A}} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12}\mathbf{e} \\ \widetilde{\mathbf{s}}_2^T \mathbf{P}_{21} & 1 - \widetilde{\mathbf{s}}_2^T \mathbf{P}_{21}\mathbf{e} \end{bmatrix} \qquad \widetilde{\boldsymbol{\alpha}}^T = \left(\widetilde{\alpha}_1, \ldots, \widetilde{\alpha}_g, \widetilde{\alpha}_{g+1} \right)$$

Approximate New PageRank Vector

$$\widetilde{\boldsymbol{\pi}}^T = \left(\widetilde{\alpha}_1, \ldots, \widetilde{\alpha}_g \,\middle|\, \widetilde{\alpha}_{g+1} \widetilde{\mathbf{s}}_2^T \right)$$

(not bad)

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

$\widetilde{\widetilde{\pi}}^T = \widetilde{\pi}^T \mathbf{P}$ (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

$\widetilde{\widetilde{\pi}}^T = \widetilde{\pi}^T \mathbf{P}$ (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

$\widetilde{\widetilde{\pi}}^T = \widetilde{\pi}^T \mathbf{P}$ (a smoothing step)

The Iterative A/D Updating Algorithm

Determine the “G-set” partition $\mathcal{S} = G \cup \overline{G}$

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

$\widetilde{\widetilde{\pi}}^T = \widetilde{\pi}^T \mathbf{P}$ (a smoothing step)

The Iterative A/D Updating Algorithm

Determine the “G-set” partition $\mathcal{S} = G \cup \overline{G}$

Approximate A/D step generates approximation $\widetilde{\pi}^T$

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

$\widetilde{\widetilde{\pi}}^T = \widetilde{\pi}^T \mathbf{P}$ (a smoothing step)

The Iterative A/D Updating Algorithm

Determine the "G-set" partition $\mathcal{S} = G \cup \overline{G}$

Approximate A/D step generates approximation $\widetilde{\pi}^T$

Smooth the result $\widetilde{\widetilde{\pi}}^T = \widetilde{\pi}^T \mathbf{P}$

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

$\tilde{\tilde{\pi}}^T = \tilde{\pi}^T \mathbf{P}$ (a smoothing step)

The Iterative A/D Updating Algorithm

Determine the "G-set" partition $\mathcal{S} = G \cup \overline{G}$

Approximate A/D step generates approximation $\tilde{\pi}^T$

Smooth the result $\tilde{\tilde{\pi}}^T = \tilde{\pi}^T \mathbf{P}$

Use $\tilde{\tilde{\pi}}^T$ as input to another approximate aggregation step

$\vdots$

Convergence

THEOREM

✓ Always converges to the new PageRank vector π^T

Convergence

THEOREM

- ✓ Always converges to the new PageRank vector π^T
- ✓ Converges for all partitions $\mathcal{S} = G \cup \overline{G}$

Convergence

THEOREM

- ✓ Always converges to the new PageRank vector π^T
- ✓ Converges for all partitions $\mathcal{S} = G \cup \overline{G}$
- ✓ Rate of convergence governed by $|\lambda_2(\mathbf{S}_2)|$

$$\mathbf{S}_2 = \mathbf{P}_{22} + \mathbf{P}_{21}(\mathbf{I} - \mathbf{P}_{11})^{-1}\mathbf{P}_{12}$$

Convergence

THEOREM

✓ Always converges to the new PageRank vector π^T

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

✓ Rate of convergence governed by $|\lambda_2(\mathbf{S}_2)|$

$$\mathbf{S}_2 = \mathbf{P}_{22} + \mathbf{P}_{21}(\mathbf{I} - \mathbf{P}_{11})^{-1}\mathbf{P}_{12}$$

THE GAME

Find a relatively small G to minimize $|\lambda_2(\mathbf{S}_2)|$

Convergence

THEOREM

- ✓ Always converges to the new PageRank vector $\boldsymbol{\pi}^T$
- ✓ Converges for all partitions $\mathcal{S} = G \cup \overline{G}$
- ✓ Rate of convergence governed by $|\lambda_2(\mathbf{S}_2)|$

$$\mathbf{S}_2 = \mathbf{P}_{22} + \mathbf{P}_{21}(\mathbf{I} - \mathbf{P}_{11})^{-1}\mathbf{P}_{12}$$

THE GAME

Find a relatively small G to minimize $|\lambda_2(\mathbf{S}_2)|$

- ✓ 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** ✧