Authoritative Sources in a Hyperlinked Environment

Jon M. Kleinberg

Presenter: Zhe Zhao

Overview

- Background and Motivation
- Approach – Authorities & Hubs
  - Construct a focused subgraph based on query
  - Computing ``hubs'' and ``authorities''
  - Iterative Algorithm and its convergence
- Expansions:
  - Similar-Page Queries
  - Multiple Set of Hubs and Authorities
- Related Work
- Conclusions

Types of Queries

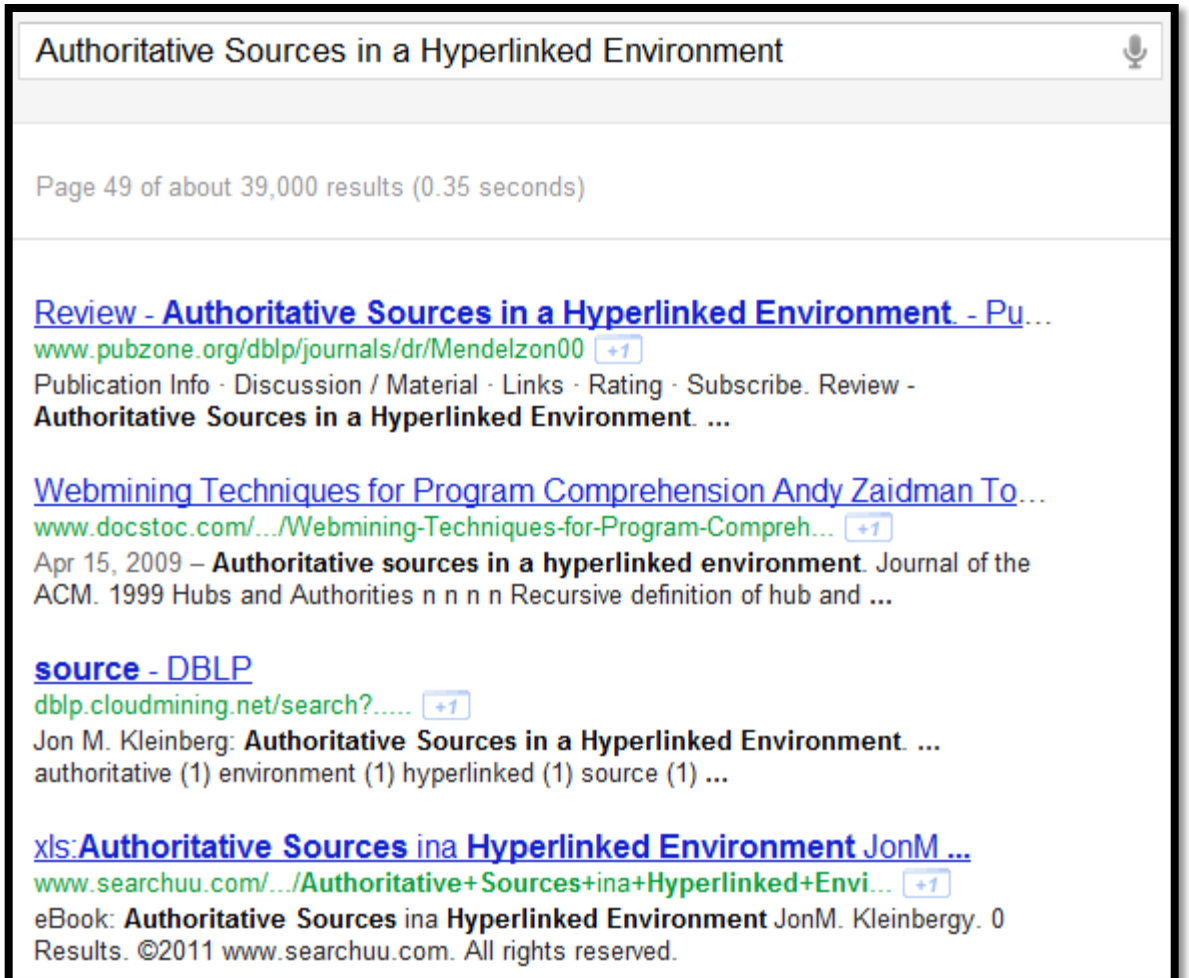
- Three Types of Queries
  - Specific queries
    - Does Netscape support the JDK 1.1 code-signing API?
  - Broad-topic queries
    - Find information about the Java programming language.
  - Similar-page queries
    - Find pages `similar‘ to java.sun.com.

Types of Queries

- Three Types of Queries
  - Specific queries
    - Does Netscape support the JDK 1.1 code-signing API?
  - Broad-topic queries **Abundance Problem!**
    - Find information about the Java programming language.
  - Similar-page queries
    - Find pages `similar‘ to java.sun.com.

Background and Motivation

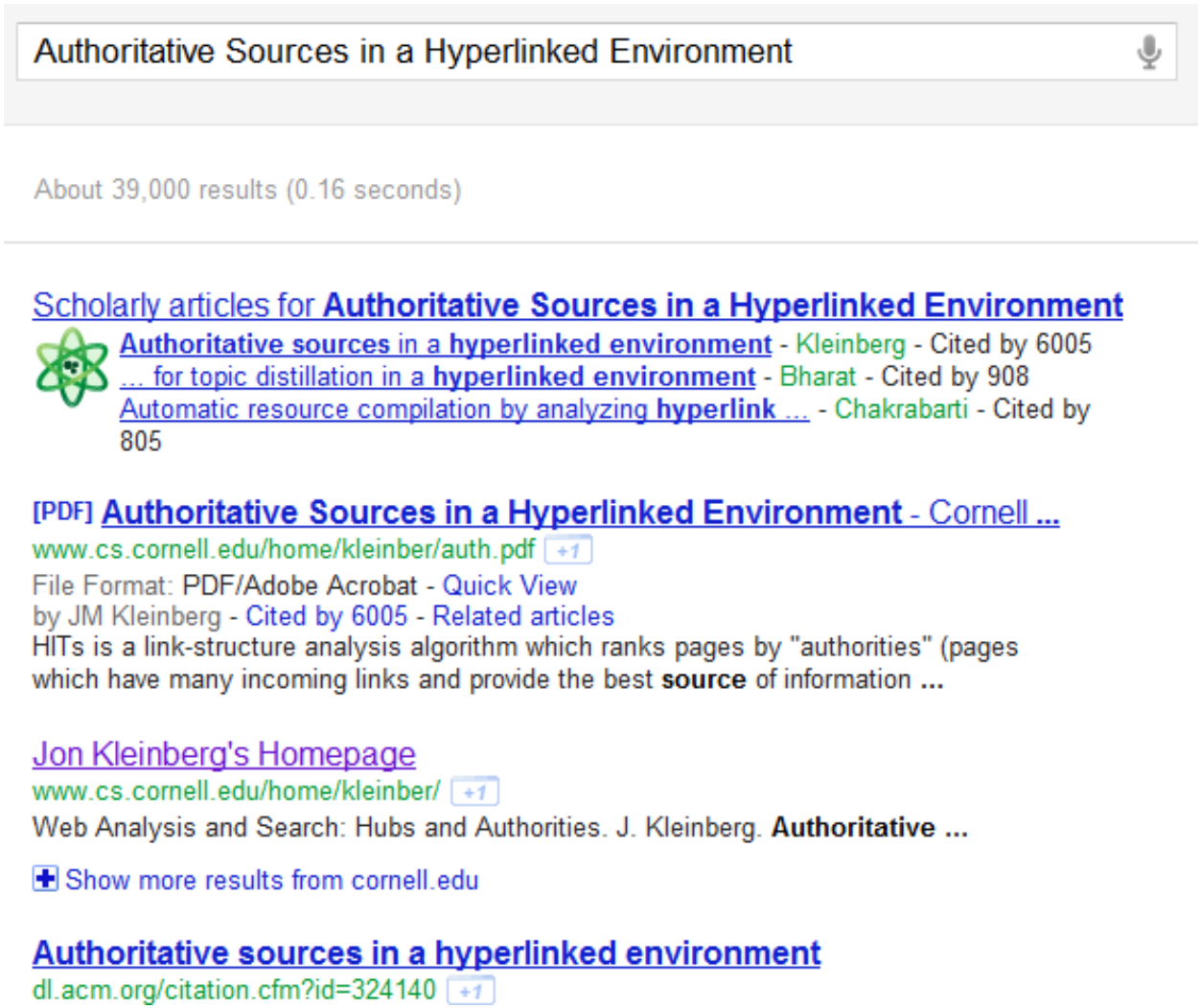
- Hard to imagine no ranking algorithms in search engine.



Ranking algorithms in web search

- Modern search engines may return millions of pages for a single query. This amount is prohibitive to preview for human users.
- Ranking algorithms will process the search results and only show the most useful information to the search engine user.

Ranking algorithms in web search



Ranking algorithms in web search

- To find a small set of most ``**authoritative**'' pages relevant to the query.
- Authority
  - Most useful/relevant/helpful results of a query.
  - ``java'' – java.com
  - ``harvard'' – harvard.edu
  - ``search engine'' – powerful search engines.

Challenge of content-based ranking

- Most useful webpage don't have the keyword
  - Query: ``Harvard''
    - 49 ``Harvard'' in www.harvard.edu
    - 357 ``Harvard'' in http://en.wikipedia.org/wiki/Harvard_University
- Pages are not sufficiently descriptive
  - ``automobile manufacturers'' in Honda or Toyota

Analysis of Link Structure

- Hyperlinks encode human latent judgment
- Reasons:
  - Navigation:
    - Back to top…
  - Relevant:
    - Webpage discussing java link to java.com
  - Popular:
    - www.yahoo.com www.google.com
  - Advertisement

Analysis of Link Structure

- Hyperlinks encode human latent judgment
- Reasons:
  - Navigation:
    - Back to top…
  - Relevant:
    - Webpage discussing java link to java.com
  - Popular:
    - www.yahoo.com www.google.com
  - Advertisement
- Some of the Reasons may be very helpful to find authoritative results.

Authorities & Hubs

- Or Hypertext-Induced Topic Search(HITS) developed by Jon Kleinberg, while visiting IBM Almaden
- IBM expanded HITS into Clever.
- Authorities
  - pages that are relevant and are linked to by many other pages
- Hubs
  - pages that link to many related authorities

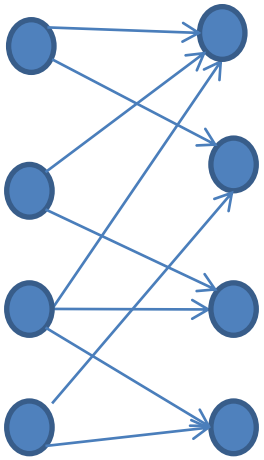
Authorities & Hubs

- Intuitive Idea to find authoritative results using link analysis:
  - Not all hyperlinks related to the conferral of authority.
  - Find the pattern authoritative pages have:
    - Authoritative Pages share considerable overlap in the sets of pages that point to them.

Authorities & Hubs

- Intuitive Idea to find authoritative results using link analysis:
  - Not all hyperlinks related to the conferral of authority.
  - Find the pattern authoritative pages have:
    - Authoritative Pages share considerable overlap in the sets of pages that point to them.

Hubs



Authorities

Authorities & Hubs

- First Step:
  - Constructing a focused subgraph of the WWW based on query
- Second Step
  - Iteratively calculate authority weight and hub weight for each page in the subgraph

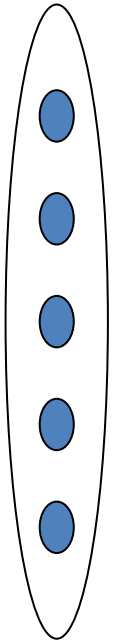
Constructing a focused subgraph

- Why not find authorities on the entire WWW?
  - The algorithm is non-trivial.
  - not necessary when there is a query.
- Objective: S_σ
  - S_σ is relatively small.
  - S_σ is rich in relevant pages.
  - S_σ contains most (or many) of the strongest authorities
- Solution:
  - Generate a Root Set Qσ from text-based search engine
  - Expand the root set

Constructing a focused subgraph

Subgraph (σ, Ɛ t,d)
σ : a query string
Ɛ *: a text-based search engine.*
t, d: natural numbers.
Let *R denote the top t results of* Ɛ *on* σ

Set *S := R*
For each page *p ε R*
  Let Γ^{+}(*p) denote the set of all pages p points to.*
  Let Γ^{-}*(p) denote the set of all pages pointing to p.*
  Add all pages in Γ^{+} (*p) to S.*
  If (Γ^{-} *(p)) < d then*
    Add all pages in Γ *(p) to S.*
  Else
    Add an arbitrary set of *d pages from* Γ^{-}*(p) to S*
End

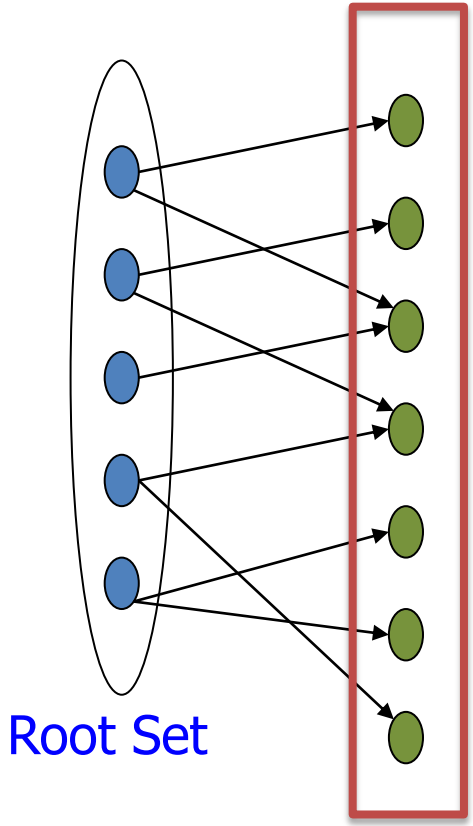


Root Set

Constructing a focused subgraph

Subgraph (σ, Ɛ t,d)
σ : a query string
Ɛ *: a text-based search engine.*
t, d: natural numbers.
Let *R denote the top t results of* Ɛ *on* σ

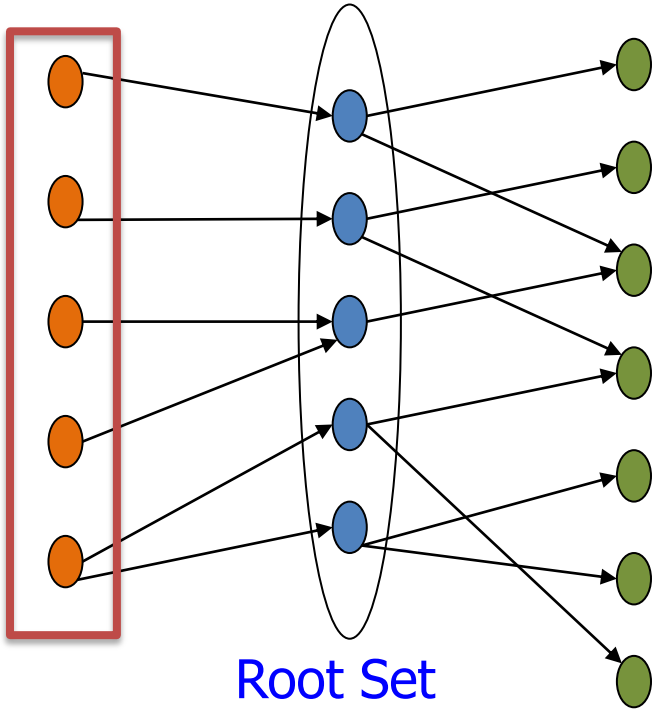
Set *S := R*
For each page *p ε R*
  Let $\Gamma^{+}(p)$ *denote the set of all pages p points to.*
  Let $\Gamma^{-}(p)$ *denote the set of all pages pointing to p.*
  Add all pages in $\Gamma^{+}(p)$ *to S.*
  if ($\Gamma^{-}(p)$) < *d then*
    Add all pages in $\Gamma(p)$ *to S.*
  Else
    Add an arbitrary set of *d pages from* $\Gamma^{-}(p)$ *to S*
End



Constructing a focused subgraph

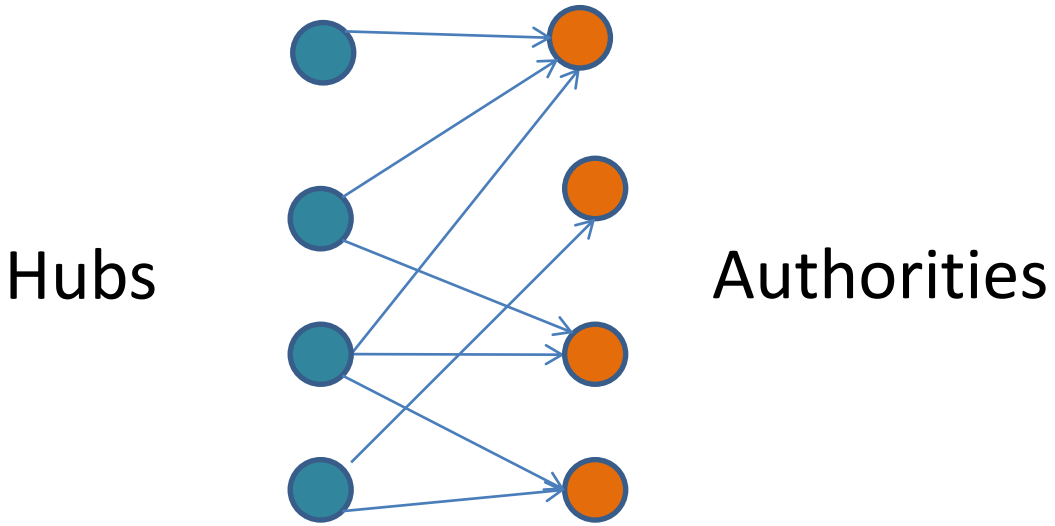
Subgraph (σ, Ɛ t,d)
σ : a query string
Ɛ *: a text-based search engine.*
t, d: natural numbers.
Let *R denote the top t results of* Ɛ *on* σ

Set *S := R*
For each page *p ε R*
  Let $\Gamma^+(p)$ *denote the set of all pages p points to.*
  Let $\Gamma^-(p)$ *denote the set of all pages pointing to p.*
  Add all pages in $\Gamma^+(p)$ *to S.*
  If ($\Gamma^-(p)$) < *d then*
    Add all pages in $\Gamma(p)$ *to S.*
  Else
    Add an arbitrary set of *d pages from* $\Gamma^-(p)$ *to S*
End



Computing Hubs and Authorities

- Rules:
  - A **good hub** points to many good authorities.
  - A **good authority** is pointed to by many good hubs.
  - Authorities and hubs have a **mutual reinforcement relationship**.



Computing Hubs and Authorities

- Let authority score of the page *i* be *x*(*i*), and the hub score of page *i* be *y*(*i*).
- mutual reinforcing relationship:
- I step:

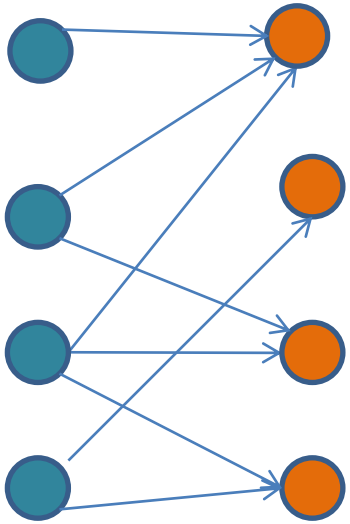
$$x(i) = \sum_{(j,i)\in E} y(j)$$

- O step:

$$y(i) = \sum_{(i,j)\in E} x(j)$$

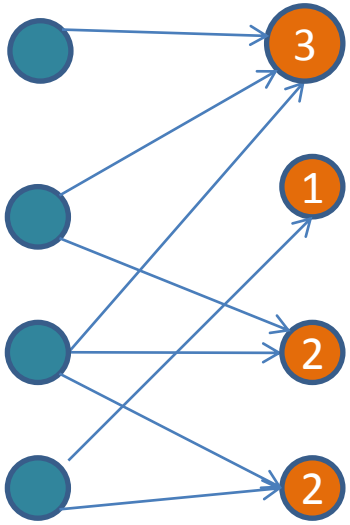
The Iterative Algorithm no normalization

- 1st Iteration



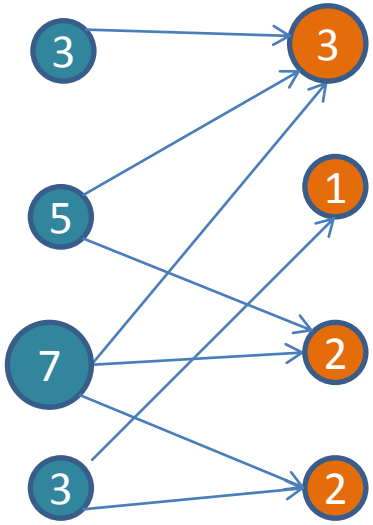
The Iterative Algorithm no normalization

- 1st Iteration
- I Step



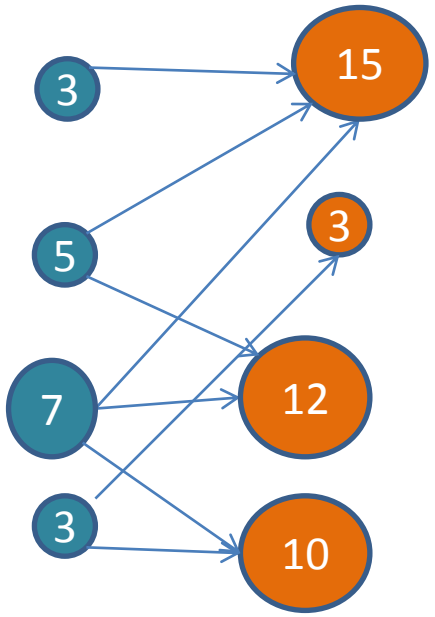
The Iterative Algorithm no normalization

- 1st Iteration
- I Step
- O Step



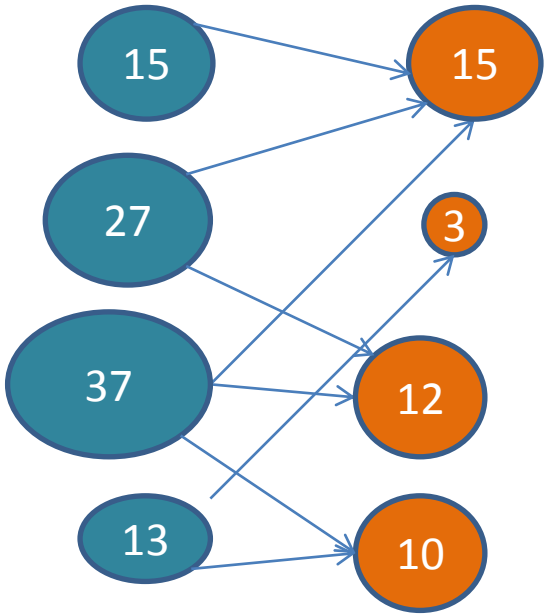
The Iterative Algorithm no normalization

- 2nd Iteration
- I Step



The Iterative Algorithm no normalization

- 2nd Iteration
- I Step
- O Step



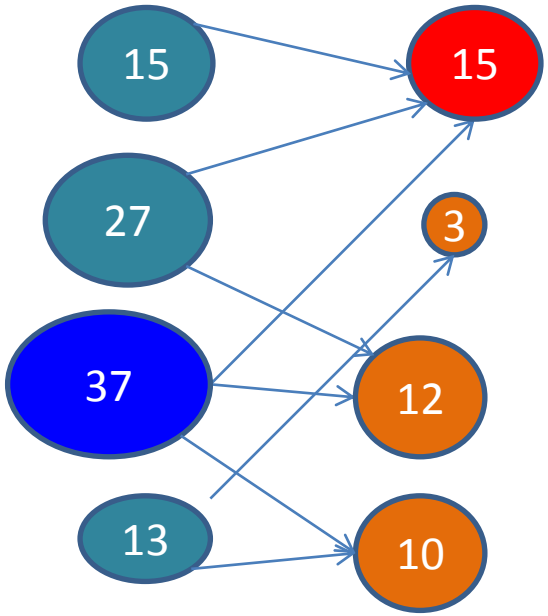
The Iterative Algorithm no normalization

- 2nd Iteration
- I Step
- O Step

...

...

...



The Iterative Algorithm

Iterate(G,k)

G: a collection of n linked pages

k: a natural number

Let z denote the vector $(1, 1, 1, \ldots, 1) \in \mathbf{R}^n$.

Set $x_0 := z$.

Set $y_0 := z$.

Initialization

For $i = 1, 2, \ldots, k$

Apply the $\mathcal{I}$ operation to (x_{i-1}, y_{i-1}), obtaining new x-weights x'_i.

Apply the $\mathcal{O}$ operation to (x'_i, y_{i-1}), obtaining new y-weights y'_i.

Normalize x'_i, obtaining x_i.

Normalize y'_i, obtaining y_i.

End

Return (x_k, y_k).

The Iterative Algorithm

Iterate(G,k)

G: a collection of n linked pages

k: a natural number

Let z denote the vector $(1, 1, 1, \ldots, 1) \in \mathbf{R}^n$.

Set $x_0 := z$.

Set $y_0 := z$.

For $i = 1, 2, \ldots, k$

I Step

Apply the $\mathcal{I}$ operation to (x_{i-1}, y_{i-1}), obtaining new x-weights x'_i.

Apply the $\mathcal{O}$ operation to (x'_i, y_{i-1}), obtaining new y-weights y'_i.

Normalize x'_i, obtaining x_i.

Normalize y'_i, obtaining y_i.

End

Return (x_k, y_k).

The Iterative Algorithm

Iterate(G,k)

G: a collection of n linked pages

k: a natural number

Let z denote the vector $(1, 1, 1, \ldots, 1) \in \mathbf{R}^n$.

Set $x_0 := z$.

Set $y_0 := z$.

For $i = 1, 2, \ldots, k$

Apply the $\mathcal{I}$ operation to (x_{i-1}, y_{i-1}), obtaining new x-weights x'_i.

Apply the $\mathcal{O}$ operation to (x'_i, y_{i-1}), obtaining new y-weights y'_i.

Normalize x'_i, obtaining x_i.

Normalize y'_i, obtaining y_i.

End

Return (x_k, y_k).

O Step

The Iterative Algorithm

Iterate(G,k)

G: a collection of n linked pages

k: a natural number

Let z denote the vector $(1, 1, 1, \ldots, 1) \in \mathbf{R}^n$.

Set $x_0 := z$.

Set $y_0 := z$.

For $i = 1, 2, \ldots, k$

Apply the $\mathcal{I}$ operation to (x_{i-1}, y_{i-1}), obtaining new x-weights x'_i.

Apply the $\mathcal{O}$ operation to (x'_i, y_{i-1}), obtaining new y-weights y'_i.

Normalize x'_i, obtaining x_i.

Normalize y'_i, obtaining y_i.

End

Return (x_k, y_k).

Normalization

Proof of Convergence

- A Matrix Perspective:
  - Denote A as adjacent matrix of the subgraph
  - I step:

$$x(i) = \sum_{(j,i)\in E} y(j) \longrightarrow x = A^T y$$

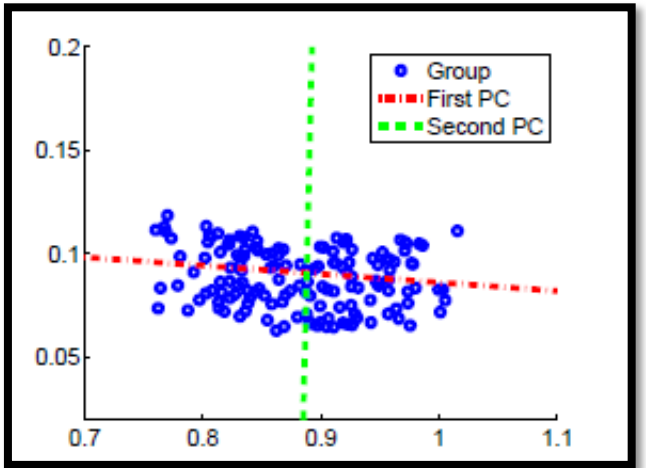
  - O step:

$$y(i) = \sum_{(i,j)\in E} x(j) \longrightarrow y = Ax$$

- Converge to eigenvector.

A Statistical View of HITS

- 1st Eigenvalue of AA^T = singular value of A
- 1st Eigenvector of AA^T = transform vector to the 1st principal component.
- Principal Component:
  - Matrix A → a set of vectors.
  - The dimension where vectors significantly distributed



A Statistical View of HITS

- The weight of authority equals the contribution of transforming the dataset to first principal component.
  - Importance of this vector for the distribution of whole dataset.
- From the statistical view:
  - HITS can be implemented by PCA
  - HITS is different from clustering using dimensionality reduction.
  - The number of samples of PCA is limited.

Example of Results:

Query “censorship” : Authorities

.378 http://www.e.org/ *EFFweb The Electronic Frontier Foundation*

.344 http://www.e.org/blueribbon.html *The Blue Ribbon Campaign for Online Free Speech*

.238 http://www.cdt.org/ *The Center for Democracy and Technology*

.235 http://www.vtw.org/ *Voters Telecommunications Watch*

.218 http://www.aclu.org/ *ACLU: American Civil Liberties Union*

Query “search engines” : Authorities

.346 http://www.yahoo.com/ *Yahoo!*

.291 http://www.excite.com/ *Excite*

.239 http://www.mckinley.com/ *Welcome to Magellan!*

.231 http://www.lycos.com/ *Lycos Home Page*

.231 http://www.altavista.digital.com/ *AltaVista: Main Page*

Expansions: Similar Page Queries

- Similar-page queries
  - Find pages `similar' to www.honda.com
- Applying HITS on Similar-Page Queries
  - Find t pages pointing to p as root set

Query "www.honda.com" : Authorities

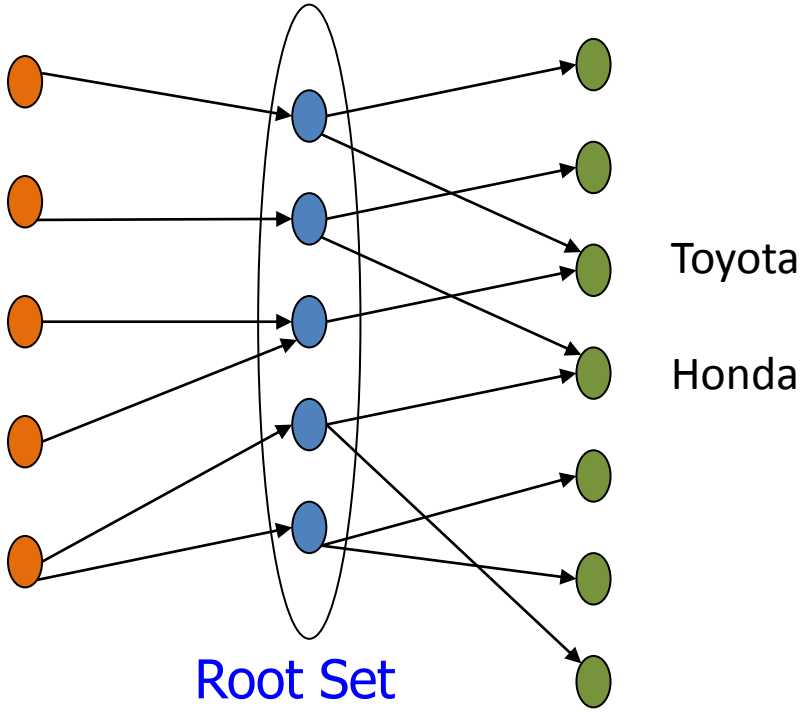
.202 http://www.toyota.com/	Welcome to @Toyota
.199 http://www.honda.com/	Honda
.192 http://www.ford.com/	Ford Motor Company
.173 http://www.bmwusa.com/	BMW of North America, Inc.
.162 http://www.volvocars.com/	VOLVO
.158 http://www.saturncars.com/	Welcome to the Saturn Web Site
.155 http://www.nissanmotors.com/	NISSAN - ENJOY THE RIDE
.145 http://www.audi.com/	Audi Homepage

Expansions: Similar Page Queries

- Why it works?
  - Does this mean that toyota.com offers a friendly hyperlink to honda.com?

Expansions: Similar Page Queries

- Why it works?
  - Does this mean that toyota.com offers a friendly hyperlink to honda.com?
  - Hubs from the root set make it possible.



Expansions: Multiple set of Hubs and Authorities

- Varies of Reasons for this:
  - The query string may have several very different meanings.
  - The current algorithm cannot find all the meanings.
    - Hubs of different meanings may not have overlap.
    - Only one type of hubs and authorities won out after iterations of mutually reinforcing.
- ``Natural'' solution:
  - Use other eigenvectors.

    -- use other principal components

Connections to Related Work

- Standing, Impact, and Influence
  - Social Network
  - Scientific Citations
- Hypertext and WWW rankings
- Clustering of Link Structures.

PageRank v.s. HITS

- PageRank
  - Computed for all web pages stored prior to the query
  - Computes authorities only
  - Fast to compute
- HITS
  - Performed on the subset generated by each query.
  - Computes authorities and hubs
  - Easy to compute, real-time execution is hard.

Which one is more suitable for large scale data set??

Conclusion

- Motivation
  - Ranking is necessary.
  - Hyperlink information is useful
- Authorities & Hubs.
  - Find authoritative pages.
  - Construct subgraph
  - Mutually reinforcing relationship
  - Iterative algorithm
- Compare to PageRank