COMP5331: Knowledge Discovery and Data Mining

Acknowledgement: Slides modified based on the slides provided by Lawrence Page, Sergey Brin, Rajeev Motwani and Terry Winograd, Jon M. Kleinberg
PageRank & HITS: Bring Order to the Web

- Background and Introduction
- Approach – PageRank
- Approach – Authorities & Hubs
Motivation and Introduction

- Why is Page Importance Rating important?
    - Huge number of web pages: 150 million by 1998
    - 1000 billion by 2008
    - Diversity of web pages: different topics, different quality, etc.

- Hard to imagine no ranking algorithms in search engine.
Motivation and Introduction

- Hard to imagine no ranking algorithms in search engine.

![Image showing search results for "Authoritative Sources in a Hyperlinked Environment"]
Motivation and Introduction

- 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.
Motivation and Introduction

Authoritative Sources in a Hyperlinked Environment

About 39,000 results (0.16 seconds)

Scholarly articles for Authoritative Sources in a Hyperlinked Environment

Authoritative sources in a hyperlinked environment - Kleinberg - Cited by 6005
... for topic distillation in a hyperlinked environment - Bharat - Cited by 908
Automatic resource compilation by analyzing hyperlink ... - Chakrabarti - Cited by 605

[PDF] Authoritative Sources in a Hyperlinked Environment - Cornell ...

www.cs.cornell.edu/home/kleinber/auth.pdf
File Format: PDF/Adobe Acrobat - Quick View
by JM Kleinberg - Cited by 6005 - Related articles
HITS is a link-structure analysis algorithm which ranks pages by "authorities" (pages which have many incoming links and provide the best source of information ...}

Jon Kleinberg's Homepage

www.cs.cornell.edu/home/kleinber/
Web Analysis and Search: Hubs and Authorities. J. Kleinberg. Authoritative ...

Show more results from cornell.edu

Authoritative sources in a hyperlinked environment
dl.acm.org/citation.cfm?id=324140
PageRank: History

- PageRank was developed by Larry Page (hence the name Page-Rank) and Sergey Brin.

- It is first as part of a research project about a new kind of search engine. That project started in 1995 and led to a functional prototype in 1998.

- Shortly after, Page and Brin founded Google.
Link Structure of the Web

- 150 million web pages → 1.7 billion links

Backlinks and Forward links:
- A and B are C’s backlinks
- C is A and B’s forward link

Intuitively, a webpage is important if it has a lot of backlinks.

What if a webpage has only one link off www.yahoo.com?
PageRank: A Simplified Version

\[ R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v} \]

- u: a web page
- \( B_u \): the set of u’s backlinks
- \( N_v \): the number of forward links of page v
- c: the normalization factor to make \( ||R||_{L1} = 1 \) (\( ||R||_{L1} = |R_1 + \ldots + R_n| \))
An example of Simplified PageRank

PageRank Calculation: first iteration

\[
\begin{bmatrix}
1/3 \\
1/2 \\
1/6
\end{bmatrix} = \begin{bmatrix}
1/2 & 1/2 & 0 \\
1/2 & 0 & 1 \\
0 & 1/2 & 0
\end{bmatrix} \begin{bmatrix}
yahoo \\
Amazon \\
Microsoft
\end{bmatrix} = \begin{bmatrix}
1/3 \\
1/3 \\
1/3
\end{bmatrix}
\]
An example of Simplified PageRank

PageRank Calculation: second iteration

\[
\begin{bmatrix}
5/12 \\
1/3 \\
1/4
\end{bmatrix} =
\begin{bmatrix}
1/2 & 1/2 & 0 \\
1/2 & 0 & 1 \\
0 & 1/2 & 0
\end{bmatrix}
\begin{bmatrix}
1/3 \\
1/3 \\
1/3
\end{bmatrix}
\]
An example of Simplified PageRank

Convergence after some iterations
A Problem with Simplified PageRank

A loop:

During each iteration, the loop accumulates rank but never distributes rank to other pages!
An example of the Problem

\[ \begin{bmatrix} 1/6 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \end{bmatrix} \]
An example of the Problem

\[
M = \begin{bmatrix}
0.5 & 0.5 & 0 \\
0.5 & 0 & 0 \\
0 & 0.5 & 1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
yahoo \\
Amazon \\
Microsoft \\
\end{bmatrix} = \begin{bmatrix}
\frac{1}{3} \\
\frac{1}{3} \\
\frac{1}{3} \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\frac{1}{4} \\
\frac{1}{6} \\
\frac{7}{12} \\
\end{bmatrix} = \begin{bmatrix}
0.5 & 0.5 & 0 \\
0.5 & 0 & 0 \\
0 & 0.5 & 1 \\
\end{bmatrix} \begin{bmatrix}
\frac{1}{3} \\
\frac{1}{6} \\
\frac{1}{2} \\
\end{bmatrix}
\]
An example of the Problem

\[ M = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \]

\[
\begin{bmatrix} \text{yahoo} \\ \text{Amazon} \\ \text{Microsoft} \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}
\]

\[
\begin{bmatrix} 5/24 \\ 1/8 \\ 2/3 \end{bmatrix} \begin{bmatrix} 1/6 \\ 5/48 \\ 35/48 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\]
Random Walks in Graphs

- The Random Surfer Model
  - The simplified model: the standing probability distribution of a random walk on the graph of the web. simply keeps clicking successive links at random

- The Modified Model
  - The modified model: the “random surfer” simply keeps clicking successive links at random, but periodically “gets bored” and jumps to a random page based on the distribution of E
Modified Version of PageRank

$$R'(u) = c_1 \sum_{v \in B_u} \frac{R'(v)}{N_v} + c_2 E(u)$$

$E(u)$: a distribution of ranks of web pages that “users” jump to when they “gets bored” after successive links at random.
An example of Modified PageRank

\[
\begin{bmatrix}
1/2 & 1/2 & 0 \\
1/2 & 0 & 0 \\
0 & 1/2 & 1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{yahoo} \\
\text{Amazon} \\
\text{Microsoft} \\
\end{bmatrix} = \begin{bmatrix}
1/3 \\
1/3 \\
1/3 \\
\end{bmatrix}
\]

\[C_1 = 0.8 \quad C_2 = 0.2\]

\[
\begin{bmatrix}
0.333 & 0.333 & 0.280 & 0.259 & 7/33 \\
0.333 & 0.200 & 0.200 & 0.179 & 5/33 \\
0.333 & 0.467 & 0.520 & 0.563 & 21/33 \\
\end{bmatrix}
\]
Dangling Links

- Links that point to any page with no outgoing links
- Most are pages that have not been downloaded yet
- Affect the model since it is not clear where their weight should be distributed
- Do not affect the ranking of any other page directly
- Can be simply removed before pagerank calculation and added back afterwards
PageRank Implementation

- Convert each URL into a unique integer and store each hyperlink in a database using the integer IDs to identify pages
- Sort the link structure by ID
- Remove all the dangling links from the database
- Make an initial assignment of ranks and start iteration
  - Choosing a good initial assignment can speed up the pagerank
- Adding the dangling links back.
Convergence Property

- PR (322 Million Links): 52 iterations
- PR (161 Million Links): 45 iterations
- Scaling factor is roughly linear in $\log n$

Convergence of PageRank Computation
Convergence Property

- The Web is an expander-like graph
  - Theory of random walk: a random walk on a graph is said to be rapidly-mixing if it quickly converges to a limiting distribution on the set of nodes in the graph. A random walk is rapidly-mixing on a graph if and only if the graph is an expander graph.

- Expander graph: every subset of nodes $S$ has a neighborhood (set of vertices accessible via outedges emanating from nodes in $S$) that is larger than some factor $\alpha$ times of $|S|$. A graph has a good expansion factor if and only if the largest eigenvalue is sufficiently larger than the second-largest eigenvalue.
Some highly accessed web pages have low page rank possibly because

- People do not want to link to these pages from their own web pages (the example in their paper is pornographic sites...)
- Some important backlinks are omitted

use usage data as a start vector for PageRank.
Hypertext-Induced Topic Search (HITS)

- 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.
Hypertext-Induced Topic Search (HITS)

- Or Authorities & Hubs, 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.

![Diagram of Hubs and Authorities](Image)
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: $S_\sigma$
  - $S_\sigma$ is relatively small.
  - $S_\sigma$ is rich in relevant pages.
  - $S_\sigma$ contains most (or many) of the strongest authorities

- Solution:
  - Generate a Root Set $Q_\sigma$ from text-based search engine
  - Expand the root set
Constructing a focused subgraph

Subgraph \((\sigma, \mathcal{E}, t, d)\)

\(\sigma\): a query string
\(\mathcal{E}\): a text-based search engine.
\(t, d\): natural numbers.
Let \(R\) denote the top \(t\) results of \(\mathcal{E}\) on \(\sigma\)

Set \(S := R\)
For each page \(p \in R\)
  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 \( (\sigma, E, t, d) \)**

\( \sigma \) : a query string  
\( E \) : a text-based search engine.  
\( t, d \) : natural numbers.  
Let \( R \) denote the top \( t \) results of \( E \) on \( \sigma \)  

Set \( S := R \)  
For each page \( p \in R \)  
- 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 ($\sigma$, $E$, $t$, $d$)

$\sigma$: a query string
$E$: a text-based search engine.
$t$, $d$: natural numbers.
Let $R$ denote the top $t$ results of $E$ on $\sigma$

Set $S := R$
For each page $p \in R$
   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:
  \[ x(i) = \sum_{(j,i) \in E} y(j) \]
- 0 step:
  \[ y(i) = \sum_{(i,j) \in E} x(j) \]
Example (no normalization)

$1^{\text{st}}$ Iteration

1 Step
Example (no normalization)

1st Iteration
1 Step
O Step
Example (no normalization)

2\textsuperscript{nd} Iteration

1 Step

\begin{center}
\begin{tikzpicture}
    \node[circle,fill=orange!50] (15) at (2,5) {15};
    \node[circle,fill=blue!50] (3) at (1,4) {3};
    \node[circle,fill=blue!50] (5) at (2,2) {5};
    \node[circle,fill=blue!50] (7) at (2,1) {7};
    \node[circle,fill=orange!50] (12) at (2,0) {12};
    \node[circle,fill=orange!50] (10) at (2,-1) {10};
    \draw[->,thick,blue] (3) -- (15);
    \draw[->,thick,blue] (5) -- (15);
    \draw[->,thick,blue] (7) -- (12);
    \draw[->,thick,blue] (3) -- (10);
\end{tikzpicture}
\end{center}
Example (no normalization)

2\textsuperscript{nd} Iteration

I Step

O Step

\begin{tikzpicture}
  \node[shape=circle,draw,fill=teal] (1) {15};
  \node[shape=circle,draw,fill=orange] (2) at (1,2) {15};
  \node[shape=circle,draw,fill=teal] (3) at (2,-2) {15};
  \node[shape=circle,draw,fill=orange] (4) at (3,-2) {15};
  \node[shape=circle,draw,fill=teal] (5) at (0,-4) {27};
  \node[shape=circle,draw,fill=orange] (6) at (1,-4) {27};
  \node[shape=circle,draw,fill=teal] (7) at (2,-4) {37};
  \node[shape=circle,draw,fill=orange] (8) at (3,-4) {37};
  \node[shape=circle,draw,fill=teal] (9) at (4,-4) {12};
  \node[shape=circle,draw,fill=orange] (10) at (5,-4) {12};
  \node[shape=circle,draw,fill=teal] (11) at (6,-4) {10};
  \node[shape=circle,draw,fill=orange] (12) at (7,-4) {10};
  \node[shape=circle,draw,fill=teal] (13) at (8,-4) {13};
  \node[shape=circle,draw,fill=orange] (14) at (9,-4) {13};

  \draw[->,thick] (1) -- (2);
  \draw[->,thick] (1) -- (3);
  \draw[->,thick] (1) -- (4);
  \draw[->,thick] (2) -- (5);
  \draw[->,thick] (2) -- (6);
  \draw[->,thick] (3) -- (7);
  \draw[->,thick] (3) -- (8);
  \draw[->,thick] (4) -- (9);
  \draw[->,thick] (4) -- (10);
  \draw[->,thick] (5) -- (11);
  \draw[->,thick] (5) -- (12);
  \draw[->,thick] (6) -- (13);
  \draw[->,thick] (6) -- (14);
\end{tikzpicture}
Example (no normalization)

- 2\textsuperscript{nd} 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 \( (1, 1, 1, \ldots, 1) \in \mathbb{R}^n \).
Set \( x_0 := z \).
Set \( y_0 := z \).

For \( i = 1, 2, \ldots, k \)

Apply the \( I \) operation to \( (x_{i-1}, y_{i-1}) \), obtaining new \( x \)-weights \( x'_i \).
Apply the \( O \) 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 \((1, 1, 1, \ldots, 1) \in \mathbb{R}^n\).
Set \(x_0 := z\).
Set \(y_0 := z\).
For \(i = 1, 2, \ldots, k\)

Apply the \(I\) operation to \((x_{i-1}, y_{i-1})\), obtaining new \(x\)-weights \(x'_i\).
Apply the \(O\) 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 \((1, 1, 1, \ldots, 1) \in \mathbb{R}^n.\)

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 \((x_k, y_k.\).\)
The Iterative Algorithm

Iterate\((G,k)\)

\(G\): a collection of \(n\) linked pages
\(k\): a natural number
Let \(z\) denote the vector \((1, 1, 1, \ldots, 1) \in \mathbb{R}^n\).  
Set \(x_0 := z\).
Set \(y_0 := z\).
For \(i = 1, 2, \ldots, k\)

\[
\begin{align*}
\text{Apply the } \mathcal{I} \text{ operation to } (x_{i-1}, y_{i-1}), \text{ obtaining new } x\text{-weights } x_i'. \\
\text{Apply the } \mathcal{O} \text{ operation to } (x_i', y_{i-1}), \text{ obtaining new } y\text{-weights } y_i'. \\
\text{Normalize } x_i', \text{ obtaining } x_i. \\
\text{Normalize } y_i', \text{ obtaining } y_i.
\end{align*}
\]

End

Return \((x_k, y_k)\).
A Statistical View of HITS

- $1^{\text{st}}$ Eigenvalue of $AA^T = $ singular value of $A$
- $1^{\text{st}}$ Eigenvector of $AA^T = $ transform vector to the $1^{\text{st}}$ 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.
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??
Summary

- PageRank is a global ranking of all web pages based on their locations in the web graph structure.
- PageRank uses information which is external to the web pages – backlinks.
- Backlinks from important pages are more significant than backlinks from average pages.
- The structure of the web graph is very useful for information retrieval tasks.

- HITS – Find authoritative pages; Construct subgraph; Mutually reinforcing relationship; Iterative algorithm.