

# Link Based Search Engines

Qiang Yang  
Simon Fraser University

4/4/01 Course Introduction 1

## Search Engine Topics

- Text-based Search Engines
  - Document based
    - Ranking: TF-IDF, Vector Space Model
    - No relationship between pages modeled
    - Cannot tell which page is important without query
  - Link-based search engines: Google, Hubs and Authorities Techniques
    - Can pick out *important* pages

4/4/01 Course Introduction 2

## The PageRank Algorithm

- Fundamental question to ask
  - What is the importance level of a page  $P$ ,  $I(P)$
- Information Retrieval
  - Cosine + TF IDF  $\rightarrow$  does not give related hyperlinks
- Link based
  - Important pages (nodes) have many other links point to it
  - Important pages also point to other important pages

4/4/01 Course Introduction 3

## The Google Crawler Algorithm

- "Efficient Crawling Through URL Ordering",
  - Junghoo Cho, Hector Garcia-Molina, Lawrence Page, Stanford
  - <http://www.www8.org>
  - <http://www-db.stanford.edu/~cho/crawler-paper/>
- "Modern Information Retrieval", BY-RN
  - Pages 380–382
- Lawrence Page, Sergey Brin. The Anatomy of a Search Engine. *The Seventh International WWW Conference (WWW 98)*. Brisbane, Australia, April 14-18, 1998.
  - <http://www.www7.org>

4/4/01 Course Introduction 4

## Back Link Metric

$IB(P)=3$

```

graph LR
    A(( )) --> P((Web Page P))
    B(( )) --> P
    C(( )) --> P
    P --> D(( ))
    P --> E(( ))
  
```

- $IB(P)$  = total number of backlinks of  $P$
- $IB(P)$  impossible to know, thus, use  $IB'(P)$  which is the number of back links crawler has seen so far

4/4/01 Course Introduction 5

## Page Rank Metric

$C=2$

```

graph LR
    T1((T1)) --> P((Web Page P))
    T2((T2)) --> P
    P --> TN((TN))
  
```

Let  $1-d$  be probability that user randomly jump to page  $P$ ;

" $d$ " is the damping factor  $d=0.9$

Let  $C_i$  be the number of out links from each  $T_i$

$$IR(P) = (1-d) + d * \sum_{i=1}^N IR(T_i) / C_i$$

4/4/01 Course Introduction 6

## Matrix Formulation

- Consider a random walk on the web (denote  $IR(P)$  by  $r(P)$ )
  - Let  $B_{ij}$  = probability of going directly from  $i$  to  $j$
  - Let  $r_i$  be the limiting probability (page rank) of being at page  $i$

$$\begin{pmatrix} b_{11} & b_{21} & \dots & b_{n1} \\ b_{12} & b_{22} & \dots & b_{n2} \\ \dots & \dots & \dots & \dots \\ b_{1n} & b_{2n} & \dots & b_{nn} \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ \dots \\ r_n \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ \dots \\ r_n \end{pmatrix} \quad B^T r = r$$

Thus, the final page rank  $r$  is a principle eigenvector of  $B^T$

4/4/01

Course Introduction

7

## How to compute page rank?

- For a given network of web pages,
  - Initialize page rank for all pages (to one)
  - Set parameter ( $d=0.90$ )
  - Iterate through the network,  $L$  times

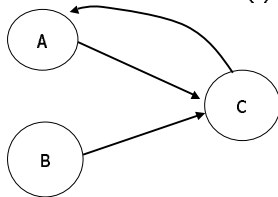
4/4/01

Course Introduction

8

## Example: iteration K=1

$IR(P) = 1/3$  for all nodes,  $d=0.9$



node	IP
A	1/3
B	1/3
C	1/3

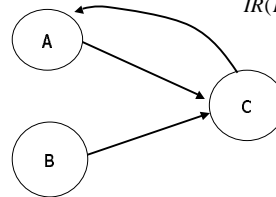
4/4/01

Course Introduction

9

## Example: k=2

$IR(P) = 0.1 + 0.9 * \sum_{i=1}^I IR(T_i) / C_i$   
 $I$  is the in-degree of  $P$



node	IP
A	0.4
B	0.1
C	0.55

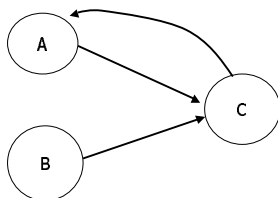
Note: A, B, C's IP values are updated in order of A, then B, then C. Use the new value of A when calculating B, etc.

4/4/01

Course Introduction

10

## Example: k=2 (normalize)



node	IP
A	0.38
B	0.095
C	0.52

4/4/01

Course Introduction

11

## Crawler Control

- All crawlers maintain several queues of URL's to pursue next
  - Google initially maintains 500 queues
  - Each queue corresponds to a web site pursuing
- Important considerations:
  - Limited buffer space
  - Limited time
  - Avoid overloading target sites
  - Avoid overloading network traffic

4/4/01

Course Introduction

12

## Crawler Control

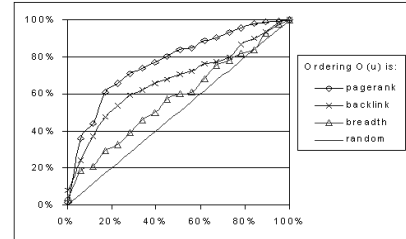
- Thus, it is important to visit important pages *first*
- Let  $G$  be a lower bound threshold on  $I(P)$
- Crawl and Stop**
  - Select only pages with  $IP > G$  to crawl,
  - Stop after crawled  $K$  pages

4/4/01

Course Introduction

13

## Test Result: 179,000 pages



Percentage of Stanford Web crawled vs.  $P_{ST}$  – the percentage of hot pages visited so far

4/4/01

Course Introduction

14

## Google Algorithm (very simplified)

- First, compute the page rank of each page on WWW
  - Query independent
- Then, in response to a query  $q$ , return pages that contain  $q$  and have highest page ranks
- A problem/feature of Google: favors big commercial sites

4/4/01

Course Introduction

15

## How powerful is Google?

- A PageRank for 26 million web pages can be computed in a few hours on a medium size workstation
- Currently has indexed a total of 1.3 Billion pages

4/4/01

Course Introduction

16

## Hubs and Authorities 1998

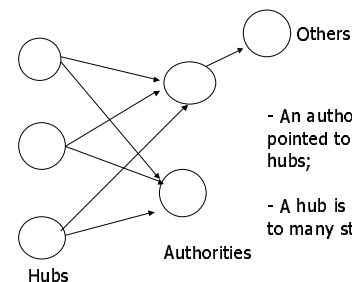
- Kleinberg, Cornell University
- <http://www.cs.cornell.edu/home/kleinber/>
- Main Idea: type "java" in a text-based search engine
  - Get 200 or so pages
  - Which one's are authoritative?
    - <http://java.sun.com>
  - What about others?
    - [www.yahoo.com/Computer/ProgramLanguages](http://www.yahoo.com/Computer/ProgramLanguages)

4/4/01

Course Introduction

17

## Hubs and Authorities



4/4/01

Course Introduction

18

## H&A Search Engine Algorithm

- First submit query  $Q$  to a text search engine
- Second, among the results returned
  - select  $\sim 200$ , find their neighbors,
  - compute *Hubs* and *Authorities*
- Third, return *Authorities* found as final result
- Important Issue*: how to find *Hubs* and *Authorities*?

4/4/01

Course Introduction

19

## Link Analysis: weights

- Let  $B_{ij}=1$  if  $i$  links to  $j$ , 0 otherwise
  - $h_i$ =hub weight of page  $i$
  - $a_i$  = authority weight of page  $i$
- Weight normalization*

$$\sum_{i=1}^N (h_i)^2 = 1 \quad (3)$$

$$\sum_{i=1}^N (a_i)^2 = 1$$

But, for simplicity, we will use

$$\sum_{i=1}^N h_i = 1 \quad (3')$$

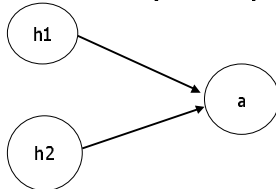
$$\sum_{i=1}^N a_i = 1$$

4/4/01

Course Introduction

20

## Link Analysis: update a-weight



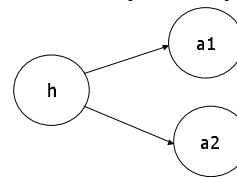
$$a_i \leftarrow \sum_{B_{ji} \neq 0} h_j = \sum B_{ji} h_j = B^T h \quad (1)$$

4/4/01

Course Introduction

21

## Link Analysis: update h-weight



$$h_i \leftarrow \sum_{B_{ij} \neq 0} a_j = \sum B_{ij} a_j = B a \quad (2)$$

4/4/01

Course Introduction

22

## H&A: algorithm

- Set value for  $K$ , the number of iterations
- Initialize all  $a$  and  $h$  weights to 1
- For  $l=1$  to  $K$ , do
  - Apply equation (1) to obtain new  $a_i$  weights
  - Apply equation (2) to obtain all new  $h_i$  weights, using the new  $a_i$  weights obtained in the last step
  - Normalize  $a_i$  and  $h_i$  weights using equation (3)

4/4/01

Course Introduction

23

## DOES it converge?

- Yes, the Kleinberg paper includes a proof
- Needs to know Linear algebra and eigenvector analysis
- We will skip the proof but only using the results:
  - The  $a$  and  $h$  weight values will converge after sufficiently large number of iterations,  $K$ .

4/4/01

Course Introduction

24

### Example: $K=1$

$h=1$  and  $a=1$  for all nodes

node	a	h
A	1	1
B	1	1
C	1	1

4/4/01 Course Introduction 25

### Example: $k=1$ (update a)

node	a	h
A	1	1
B	0	1
C	2	1

4/4/01 Course Introduction 26

### Example: $k=1$ (update h)

node	a	h
A	1	2
B	0	2
C	2	1

4/4/01 Course Introduction 27

### Example: $k=1$ (normalize)

Use Equation (3')

node	a	h
A	1/3	2/5
B	0	2/5
C	2/3	1/5

4/4/01 Course Introduction 28

### Example: $k=2$ (update a, h, normalize)

Use Equation (1)

node	a	h
A	1/5	4/9
B	0	4/9
C	4/5	1/9

If we choose a threshold of  $1/2$ , then C is an Authority, and there are no hubs.

4/4/01 Course Introduction 29

### Search Engine Using H&A

- For each query  $q$ ,
  - Enter  $q$  into a text-based search engine
  - Find the top 200 pages
  - Find the neighbors of the 200 pages by one link, let the set be  $S$
  - Find hubs and authorities in  $S$
  - Return authorities as final result

4/4/01 Course Introduction 30

## ■ Conclusions

- Link based analysis is very powerful in find out the important pages
- Models the web as a graph, and based on in-degree and out-degree
- Google: crawl only important pages
- H&A: post analysis of search result