IR: Information Retrieval

FIB, Master in Innovation and Research in Informatics

Slides by Marta Arias, José Luis Balcázar,
Ramon Ferrer-i-Cancho, Ricard Gavaldá
Department of Computer Science, UPC

Fall 2018

http://www.cs.upc.edu/~ir-miri

5. Web Search. Architecture of simple IR systems

Searching the Web, I

When documents are interconnected

The World Wide Web is huge

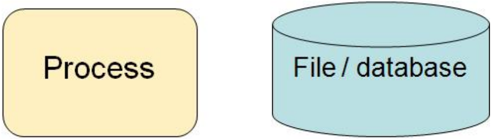
- 100,000 indexed pages in 1994
- 10,000,000,000's indexed pages in 2013

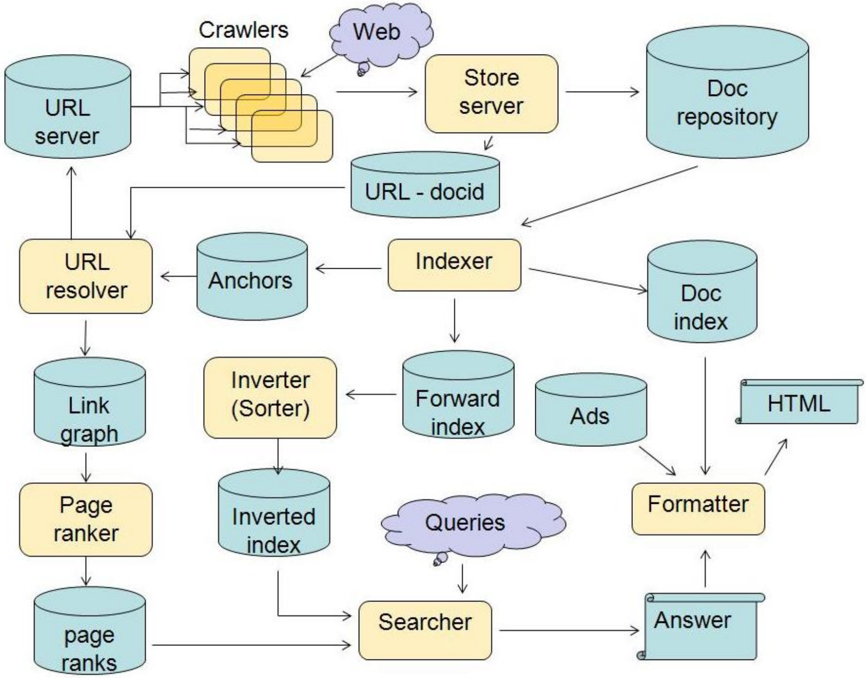
- Most queries will return *millions* of pages with high similarity.
- Content (text) alone cannot discriminate.
- Use the structure of the Web - a graph.
- Gives indications of the prestige - usefulness of each page.

How Google worked in 1998

S. Brin, L. Page: “The Anatomy of a Large-Scale Hypertextual Web Search Engine”, 1998

Notation:




Crawlers
Web
URL server
Store server
Doc repository
URL - docid
URL resolver
Anchors
Indexer
Doc index
Link graph
Inverter (Sorter)
Forward index
Ads
HTML
Page ranker
Inverted index
Queries
Formatter
page ranks
Searcher
Answer

Some components

- URL store: URLs awaiting exploration
- Doc repository: full documents, zipped
- Indexer: Parses pages, separates text (to Forward Index), links (to Anchors) and essential text info (to Doc Index)
  - Text in an anchor very relevant for *target* page

    `<a href="http://page">anchor</a>`

  - Font, placement in page makes some terms extra relevant
- Forward index: docid $\rightarrow$ list of terms appearing in docid
- Inverted index: term $\rightarrow$ list of docid's containing term

The inverter (sorter), I

Transforms forward index to inverted index

First idea:

```
for every entry document d
    for every term t in d
        add docid(d) at end of list for t;
```

Lousy locality, many disk seeks, too slow

The inverter (sorter), II

Better idea for indexing:

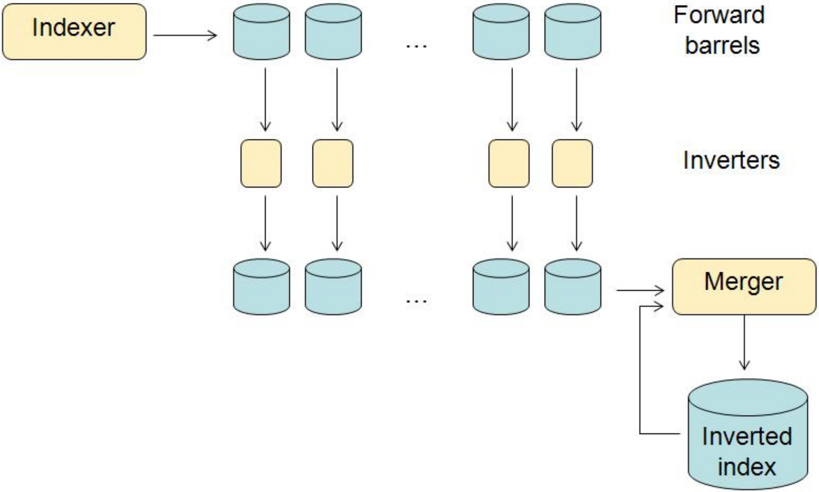
```
create in disk an empty inverted file, ID;
create in RAM an empty index IR;
for every document d
    for every term t in d
        add docid(d) at end of list for t in IR;
        if RAM full
           for each t, merge the list for t in IR
               into the list for t in ID;
```

Merging previously sorted lists is sequential access

Much better locality. Much fewer disk seeks.

The inverter (sorter), III

The above can be done concurrently on different sets of documents:



The inverter (sorter), IV

- Indexer ships barrels, fragments of forward index
- Barrel size = what fits in main memory
- Separately, concurrently inverted in main memory
- Inverted barrels merged to inverted index
- 1 day instead of estimated months

Searching the Web, I

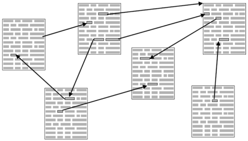
When documents are interconnected

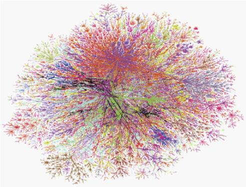
The internet is huge

- 100,000 indexed pages in 1994
- 10,000,000,000 indexed pages at end of 2011

To find content, it is necessary to search for it

- We know how to deal with the content of the webpages
- But.. what can we do with the structure of the internet?





Searching the Web, II

Meaning of a hyperlink

When page A links to page B, this means

- A's author thinks that B's content is interesting or important
- So a link from A to B, adds to B's reputation

But not all links are equal..

- If A is very important, then $A \to B$ "counts more"
- If A is not important, then $A \to B$ "counts less"

In today's lecture we'll see two algorithms based on this idea

- *Pagerank* (Brin and Page, oct. 98)
- *HITS* (Kleinberg, apr. 98)

Pagerank, I

The idea that made Google great

Intuition:

$\boxed{\text{A page is important if it is pointed to by other important pages}}$

- Circular definition ...
- not a problem!

Pagerank, II

Definitions

The web is a graph $G = (V, E)$

- $V = \{1, .., n\}$ are the nodes (that is, the pages)
- $(i, j) \in E$ if page i points to page j
- we associate to each page i, a real value p_i (i's *pagerank*)
- we impose that $\sum_{i=1}^{n} p_i = 1$

How are the p_i's related

- p_i depends on the values p_j of pages j pointing to i

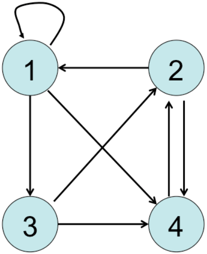
$$\boxed{p_i = \sum_{j \to i} \frac{p_j}{out(j)}}$$

- where $out(j)$ is j's *outdegree*

Pagerank, III

Example

A set of $n + 1$ linear equations:



$$p_i = \sum_{j \to i} \frac{p_j}{out(j)}$$

$$\begin{aligned}
p_1 &= \frac{p_1}{3} + \frac{p_2}{2} \\
p_2 &= \frac{p_3}{2} + p_4 \\
p_3 &= \frac{p_1}{3} \\
p_4 &= \frac{p_1}{3} + \frac{p_2}{2} + \frac{p_3}{2} \\
1 &= p_1 + p_2 + p_3 + p_4
\end{aligned}$$

Whose solutions is:

$p_1 = 6/23, p_2 = 8/23, p_3 = 2/23, p_4 = 7/23$

Pagerank, IV
Formally

Equations

- $p_i = \sum_{j:(j,i)\in E} \frac{p_j}{out(j)}$ for each $i \in V$
- $\sum_{i=1}^{n} p_i = 1$

where $out(i) = |\{j : (i,j) \in E\}|$ is the *outdegree* of node i

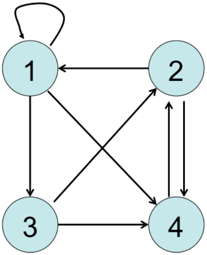
If $|V| = n$

- $n + 1$ equations
- n unknowns

Could be solved, for example, using Gaussian elimination in time $O(n^3)$

Pagerank, V

Example, revisited



A set of linear equations:

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{pmatrix}$$

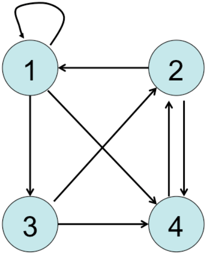
namely: $\vec{p} = M^T \vec{p}$ and additionally $\sum_i p_i = 1$

Whose solutions is:

$\vec{p}$ is the eigenvector of matrix M^T associated to eigenvalue 1

Pagerank, VI

Example, revisited



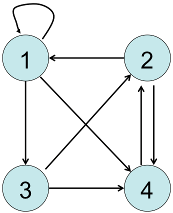
What does M^T look like?

$$M^T = \begin{pmatrix} \frac{1}{3} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}$$

M^T is the *transpose* of the row-normalized adjacency matrix of the graph !

Pagerank, VII

Example, revisited



Adjacency matrix

$$A = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$M = \begin{pmatrix} 1/3 & 0 & 1/3 & 1/3 \\ 1/2 & 0 & 0 & 1/2 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

(rows add up to 1)

$$M^T = \begin{pmatrix} 1/3 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 1 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1/2 & 1/2 & 0 \end{pmatrix}$$

(columns add up to 1)

Pagerank, VIII

Example, revisited

$$A = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad M = \begin{pmatrix} \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad M^T = \begin{pmatrix} \frac{1}{3} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}$$

Question:

Why do we need to row-normalize and transpose A?

Answer:

- *Row normalization*: because $p_i = \sum_{j:(j,i) \in E} \frac{p_j}{out(j)}$
- *Transpose*: because $p_i = \sum_{j:(j,i) \in E} \frac{p_j}{out(j)}$, that is,

  p_i depends on i's *incoming edges*

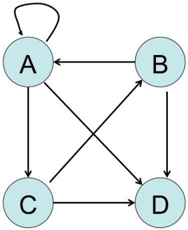
Pagerank, IX

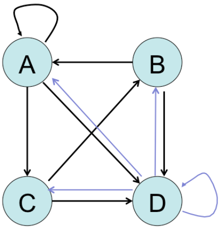
It is just about solving a system of linear equations!

.. but

- How do we know a solution exists?
- How do we know it has a single solution?
- How can we compute it efficiently?

For example, the graph on the left has no solution.. (check it!)
but the one on the right does





Pagerank, X

How do we know a solution exists?

Luckily, we have some results from *linear algebra*

Definition

A matrix M is stochastic, if

- All entries are in the range $[0, 1]$
- Each row adds up to 1 (i.e., M is row normalized)

Theorem (Perron-Frobenius)

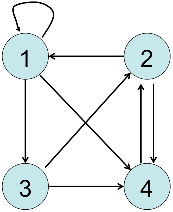
If M is stochastic, then it has at least one stationary vector, i.e., one non-zero vector p such that

$$M^T p = p.$$

Pagerank, XI

Equivalently: the random surfer view

Now assume M is the transition probability matrix between states in G



$$M = \begin{pmatrix} 1/3 & 0 & 1/3 & 1/3 \\ 1/2 & 0 & 0 & 1/2 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Let $\vec{p}(t)$ be the probability over states at time t

- E.g., $p_j(0)$ is the probability of being at state j at time 0

Random surfer jumps from page i to page j with probability m_{ij}

- E.g., probability of transitioning from state 2 to state 4 is $m_{24} = 1/2$

Pagerank, XII

The random surfer view

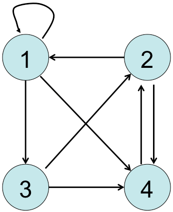
- Surfer starts at random page according to probability distribution $\vec{p}(0)$
- At time $t > 0$, random surfer follows one of current page's links uniformly at random

$$\vec{p}(t) := M^T \vec{p}(t-1)$$

- In the limit $t \to \infty$:
  - $\vec{p}(t) = \vec{p}(t+1) = \vec{p}(t+2) = .. = \vec{p}$
  - so $\vec{p}(t) = M^T \vec{p}(t-1)$
  - $\vec{p}(t)$ converges to a solution p s.t. $p = M^T p$ (the pagerank solution)!

Pagerank, XIII

Random surfer example



$$M^T = \begin{pmatrix} \frac{1}{3} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}$$

- $\vec{p}(0)^T = (1, 0, 0, 0)$
- $\vec{p}(1)^T = (1/3, 0, 1/3, 1/3)$
- $\vec{p}(2)^T = (0.11, 0.50, 0.11, 0.28)$
- ..
- $\vec{p}(10)^T = (0.26, 0.35, 0.09, 0.30)$
- $\vec{p}(11)^T = (0.26, 0.35, 0.09, 0.30)$

Pagerank, XIV

An algorithm to solve the eigenvector problem (find p s.t. $p = M^T p$)

The Power Method

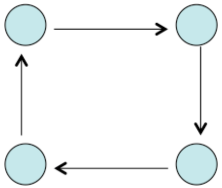
- Chose initial vector $\vec{p}(0)$ randomly
- Repeat $\vec{p}(t) \leftarrow M^T \vec{p}(t-1)$
- Until convergence (i.e. $\vec{p}(t) \approx \vec{p}(t-1)$)

We are hoping that

- The method converges
- The method converges fast
- The method converges fast to the pagerank solution
- The method converges fast to the pagerank solution regardless of the initial vector

Pagerank, XV

Convergence of the Power method: aperiodicity required



Try out the power method with $\vec{p}(0)$:

$$\begin{pmatrix}1/4\\1/4\\1/4\\1/4\end{pmatrix}, \text{ or } \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \text{ or } \begin{pmatrix}1/2\\0\\1/2\\0\end{pmatrix}$$

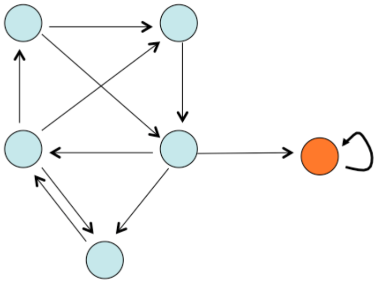
Not being able to break the cycle looks problematic!

- .. so will require graphs to be aperiodic
  - no integer $k > 1$ dividing the length of every cycle

Pagerank, XVI

Convergence of the Power method: strong connectedness required

What happens with the pagerank in this graph?



The sink hoards all the pagerank!

- need a way to leave sinks
- .. so we will force graphs to be strongly connected

Pagerank, XVII

A useful theorem from Markov chain theory

Theorem

If a matrix M is strongly connected and aperiodic, then:

- $M^T\vec{p} = \vec{p}$ *has exactly one non-zero solution such that* $\sum_i p_i = 1$
- *1 is the largest eigenvalue of* M^T
- *the Power method converges to the* $\vec{p}$ *satisfying* $M^T\vec{p} = \vec{p}$*, from any initial non-zero* $\vec{p}(0)$
- *Furthermore, we have exponential fast convergence*

To guarantee a solution, we will make sure that the matrices that we work with are strongly connected and aperiodic

Pagerank, XVIII

Guaranteeing aperiodicity and strong connectedness

Definition (The Google Matrix)

Given a damping factor λ such that: $0 < \lambda < 1$:

$$G = \lambda M + (1 - \lambda)\frac{1}{n}J$$

where J is a $n \times n$ matrix containing 1 in each entry

Observe that:

- G is stochastic
  - .. because G is a weighted average of M and $\frac{1}{n}J$, which are also stochastic
- for each integer $k > 0$, there is a non-zero probablity path of length k from every state to any other state of G
  - .. implying that G is strongly connected and aperiodic
- and so the Power method will converge on G, and fast!

Pagerank, XIX

Teleportation in the random surfer view

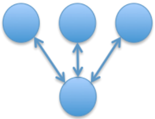
The meaning of λ

- With probability λ, the random surfer follows a link in current page
- With probability $1 - \lambda$, the random surfer jumps to a random page in the graph (teleportation)

Pagerank, XX

Excercise, I

Compute the pagerank value of each node of the following graph assuming a damping factor $\lambda = 2/3$:



Hint: solve the following system, using $p_2 = p_3 = p_4$

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{pmatrix} = \left[\frac{2}{3} \begin{pmatrix} 0 & 1 & 1 & 1 \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 & 0 \end{pmatrix} + \frac{1}{3} \cdot \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \right] \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{pmatrix}$$

Pagerank, XXI

Exercise, II

Compute the pagerank vector $\vec{p}$ of graph with row-normalized matrix M for damping factor λ in closed matrix form.

Answer:

$$\vec{p} = (I - \lambda M^T)^{-1} \begin{pmatrix} \frac{1-\lambda}{n} \\ \vdots \\ \frac{1-\lambda}{n} \end{pmatrix}$$

Topic-sensitive Pagerank, I

Observe that pageranks are independent of user's query

- Advantages
  - Computed off-line
  - Collective reputation
- Disadvantages
  - Insensitive to particular user's needs

Topic-sensitive Pagerank, II

Assume there is a small set of K topics (sports, science, politics, ...)

- Each topic $k \in \{1, .., K\}$ is defined by a subset of the web pages T_k
- For each k, compute pagerank of node i for topic k:

  $p_{i,k} =$ “pagerank of node i with teleportation reduced to T_k”

- Finally compute ranking score of a page i given query q

$$score(i, q) = \sum_{k=1}^{K} sim(T_k, q) \cdot p_{i,k}$$

HITS, I
Hypertext Induced Text Search

Interest of a web page due to two different reasons

- page content is insteresting (*authority*), or
- page points to interesting pages (*hub*)

HITS main rationale

- hubs are important if they point to important authorities
- authorities are impotant if pointed to by important hubs
- .. but .. circular definition again .... not a problem!

HITS, II

Definition of authority and hub value (a_i and h_i)

Associate to each page i an authority value a_i and a hub value h_i

- vector of all authority values is $\vec{a}$
- vector of all hub values is $\vec{h}$

Keep these vectors normalized (notice L2 norm!)

- $\|\vec{a}\| = \sum_i a_i^2 = 1$, and $\|\vec{h}\| = \sum_i h_i^2 = 1$

For appropriate scaling constants c and d

- $a_i = c \cdot \sum_{j \to i} h_j$, and $h_i = d \cdot \sum_{i \to j} a_j$

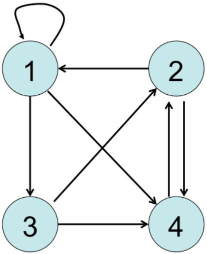
Notice not a linear system anymore!

- ... but still ok with a variant of the power method

HITS, III

Example

Our old graph



Adjacency matrix

$$A = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

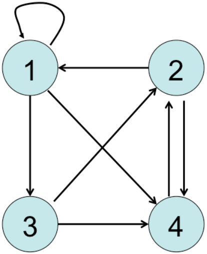
$a_1 = c \cdot (h_1 + h_2)$ *// here we use A's first column*

$$a_1 \propto (1, 1, 0, 0) \cdot \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{pmatrix} = (1, 1, 0, 0) \cdot \vec{h}$$

HITS, IV

Example

Our old graph



Adjacency matrix

$$A = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$h_2 = d \cdot (a_1 + a_4)$ // *here we use A's second row*

$$h_2 \propto (1, 0, 0, 1) \cdot \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} = (1, 0, 0, 1) \cdot \vec{a}$$

HITS, V

Update rule for $\vec{a}$ and $\vec{h}$

Written in compact matrix form

- To update authority values
  - $\vec{a} := A^T \cdot \vec{h}$
  - normalize afterwards $\vec{a} := \frac{\vec{a}}{\|a\|}$ so that $\|a\| = 1$
- To update hub values
  - $\vec{h} := A \cdot \vec{a}$
  - normalize afterwards $\vec{h} := \frac{\vec{h}}{\|h\|}$ so that $\|h\| = 1$

HITS, VI

The power method for finding $\vec{a}$ and $\vec{h}$

Given adjancecy matrix A

- Initialize $\vec{a} = \vec{h} = (1, 1, .., 1)^T$
- Normalize $\vec{a}$ and $\vec{h}$ so that $\|a\| = \|h\| = 1$
- Repeat until convergence
  - $\vec{a} := A^T \cdot \vec{h}$
  - normalize $\vec{a}$ so that $\|a\| = 1$
  - $\vec{h} := A \cdot \vec{a}$
  - normalize $\vec{h}$ so that $\|h\| = 1$

HITS, VII
HITS algorithm

Query answering algorithm HITS

- Get query q and run content-based searcher on q
- Let $RootSet$ be the top-k ranked pages
- Expand pages to $BaseSet$ by adding all pages pointed to and by pages in $RootSet$
- compute hub and authority values for the subgraph of web induced by $BaseSet$
- Rank pages in $BaseSet$ according to $\vec{a}$, $\vec{h}$, and content

HITS, VIII

HITS algorithm illustrated

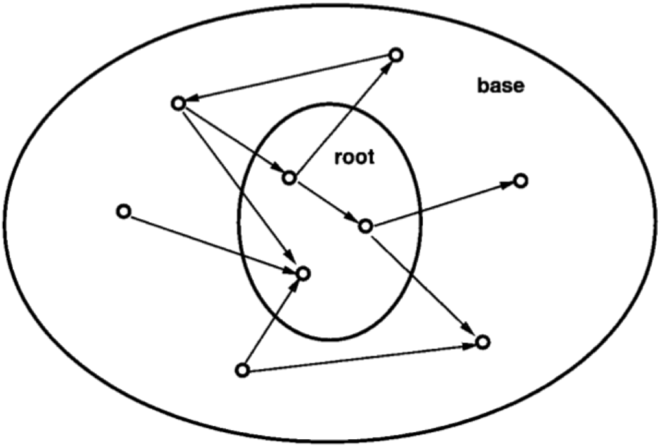


FIG. 1. Expanding the root set into a base set.

HITS vs. Pagerank

Pros of HITS vs. Pagerank

- Sensitive to user queries

Cons of HITS vs. Pagerank

- Compute online, not offline!
- More vulnerable to webspamming