Artificial Intelligence Applied to the Web

Chapter 3 Part 2 - Web Page Rank Algorithms

Diego Cornejo, Felipe Hernández and Juan Velásquez

University of Chile
Departament of Industrial Engineering

Spring 2025

Outline

PageRank

Idea

- An Hyperlink that point to me works like a *recommendation* from other page.
- With more recommendation, more **important** is my page.
- The more important are my recommenders, the more **important** is my page.

Rank

A **numeric value** to represent the importance of a page.

We aren't inspecting the content. We are looking for recommendations

PageRank

Probabilistic point of view.

- We can view the importance of a web page i as the probability that a random surfer on the Internet opens a browser to any page and starts following **hyperlinks** visits i.
- Weights on edges in a transition matrix could be assigned in a *probabilistic way*.
- We can model the process as a **random walk** on graphs. Each page has equal probability $\frac{1}{n}$ to be chosen as a starting point.
- Then, probability that page i is visited after one step is equal to Ax and so on.
- The probability that page i will be visited after k steps is equal to $A^k x$.
- That sequence converges to a **unique probabilistic vector** v^* called the *stationary distribution*.
- **This will be our PageRank**.

PageRank

Mathematical foundation

- $\Pi = (r_i)$ The vector of Rank number, the rank r_i for page p_i
- **Each page that point to me, add a fraction of its own rank for my total rank**
- That's means it is a **linear combination**: $r_i = \sum_k T_{ik} r_{ik}$
- $T = (t_{ik})$ The transition matrix; and it is stochastic: $\sum_k t_{ik} = 1$ (the total rank is conserved).
- **What can we do with this?**

PageRank

Mathematical foundation

Perron-Frobenius Theorem

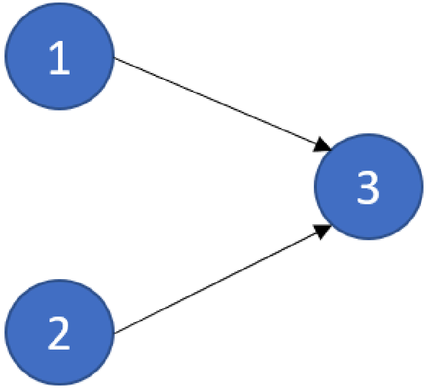
- Is about the **eigenvectors** and **eigenvalues** of *non-negative and irreducible matrices.*
- A matrix is called *non-negative* if all of its entries are ≥ 0.
- A matrix is called *irreducible* if for any of its entries (i, j) there is k such that the (i, j) entry of A^k is **positive**.
- It says that for A a non-negative and irreducible matrix, it is an *eigenvalue* λ_{max} and for all other eigenvalues λ we have $|\lambda| \leq \lambda_{\text{max}}$.
- Moreover, if the sum of all entries of each column of A is 1, then $\lambda_{\text{max}} = 1$.

But, there is a little problem: **the matrices we obtain from the Web are not always irreducible**

PageRank

Mathematical foundation - Issue 1

There could be: **nodes with no outgoing edges**, also called **dangling nodes**.



$$v_0 = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}$$

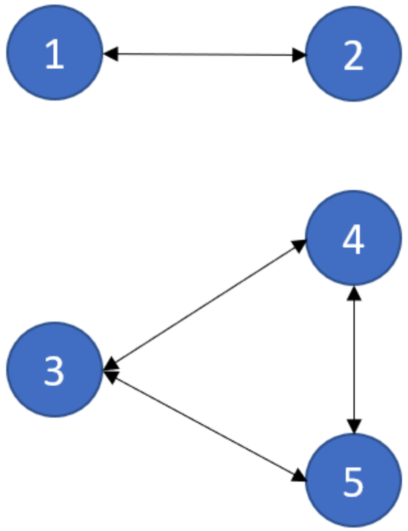
$$v_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{2}{3} \end{bmatrix}$$

$$v_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ \frac{2}{3} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

PageRank

Mathematical foundation - Issue 2

There could be: **disconnected components**.



$$A = \left[\begin{array}{cc|ccc} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{array}\right]$$

$$v = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad u = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Both eigenvectors with eigenvalue = 1

PageRank

Mathematical foundation - The solution of Page and Brin

- They propose a fix positive constant d between 0 and 1 called the **dampling factor** (with a typical value $d = 0.85$).
- This models the random surfer behaviour!
- Now, the *transition matrix* used to compute PageRank will be a new matrix M such that:

$$M = dT + \frac{(1-d)}{n}E$$

  with E as a matrix of ones.
- M is called the **Google matrix**.

PageRank

PageRank Algorithm

The assumption is the **importance of a page** is given for the **importance of the pages that pointed it**.

$$r_p^{(k+1)} = (1-d)\frac{1}{n}r_p^{(k)} + d\sum_{\forall q,p\in P\ |\ q\to p}\frac{1}{N_q}r_q^{(k)}$$

PageRank

PageRank Algorithm

$$r_p^{(k+1)} = (1-d)\frac{1}{n}r_p^{(k)} + d \sum_{\forall q,p \in P \mid q \to p} \frac{1}{N_q} r_q^{(k)}$$

Where:

- r_p: Importance of page p
- n: Number of pages in the web graph
- d: Probability that the surfer follows some out-links of q when visit that page
- N_q: Number of out-links from page q
- r_q: Importance of page q
- $\frac{1}{N_q}$: Conditional probability of going to another page

The PageRank is the fixed point value of this recurrence!

PageRank

PageRank Algorithm

In matrix form:

$$\Pi^{(k+1)} = \left(dT + \left(\frac{1-d}{n}\right)E\right)\Pi^{(k)}$$

- d: The probability of going out the n node
- T: An transition matrix (stochastic) that is interpreted as the transition probability. But in the Google way they consider **equiprobability** $\frac{1}{N_q}$
- E: Is the *1 matrix*. A matrix filled with 1.

Basically, replace the usual transition matrix with the Google matrix and compute the eigenvector with eigenvalue equal to 1

PageRank

PageRank Algorithm

Intuitive Justification

- Thought as a **model of user behavior**.
- **Random surfer** that keep clicking on links **starting from one random page**, and **some time get bored** and **continue** from other random page.
- **PageRank** value correspond to the **probability that the random surfer visit the page**.
  - The **stationary probability.**
- Then the **d parameter** correspond to the **probability to start again from another page**.

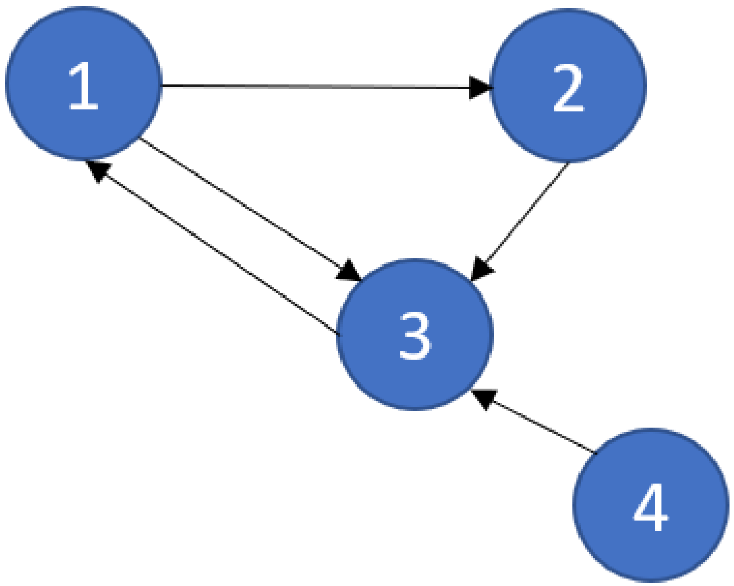
PageRank

PageRank Algorithm

1. Initialize: $\Pi^{(0)} = \left(\frac{1}{n}, \ldots, \frac{1}{n}\right)$
2. Set d, usually $d = 0.85$
3. Calculate: $\Pi^{(k+1)} = \left(dT + \frac{(1-d)}{n}E\right)\Pi^{(k)}$
4. If $\left\|\Pi^{(k+1)} - \Pi^{(k)}\right\| < \xi$ stop and return. Else, $\Pi^{(k)} = \Pi^{(k+1)}$ and go to the point 3.

PageRank

PageRank Algorithm - Example



Considering the graph above, get its PageRank

PageRank

PageRank Algorithm

Disadvantages

- If a page is pointed by another one, it means that the page receives a vote for the PageRank calculus.
- If a page is pointed by a lot of pages, it means that the page is important.
- **Only the good pages are pointed by others one**, but:
  - **Reciprocal link**: If the page A links page B, then page B will link page A.
  - **Link Requirements**: Some web pages give electronic gifts, like programs, documents etc., if another page points it.
  - **Near persons community**: For instance, friends and relatives that from their pages point another friend or relative only because of the human relationship between them.

PageRank

PageRank Algorithm

- **Not the real Google algorithm.** It is a very carefully hidden secret.
- The original PageRank doesn't consider text content (keywords) on the page. But the real one **does**. The real algorithm also considers **n-grams**
- The real algorithm also considers user behavior. They capture it with:
  - Click on links
  - Google toolbar
  - Google web-accelerator (a Google proxy)
  - Gmail and Youtube

PageRank

PageRank sensibility and Link spammers

- Could a group of people artificially reference pages between them in order to increase the rank?
- A **parallel algorithm search for spammer** and lowers its rank.
- The sensibility from a spamming update is:

$$\left\|\Pi - \tilde{\Pi}\right\|_1 < \frac{2d}{1-d}\sum_{i\in U} r_i$$

  - U: A community of spammers
  - Π: Change on each $i \in U$, r_i are the original rank
  - The rank is bound nevertheless any action of a group U of spammers

Outline

Page Rank

HITS

SALSA

HITS Algorithm

HITS Algorithm Introduction

- Proposed by John Kleinberg in 1998.
- Stands for Hypertext Induced Topic Search.
- Expand the list of relevant pages returned by a search engine.
- Produce two rankings: Authority ranking and Hub ranking.

Assumptions

- A credible page will point to credible pages.
- Credible pages are pointed by others.

The page ranking depends on the user query and the hyperlink structure that follows from paths of the most credible pages

HITS Algorithm

Authority and Hubs Pages

Definitions

- Authority: A page with many in-links.
- Hub: A page with many out-links.

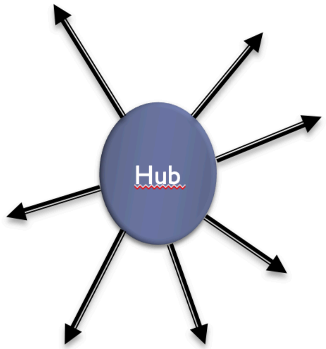
The idea is that:

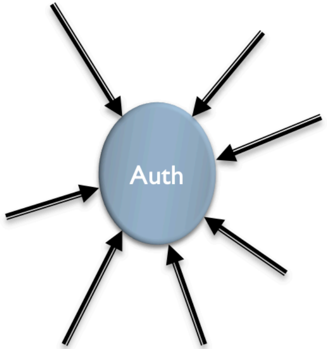
- A page may have authoritative content on some topic and thus many people trust it and thus link to it.
- When a user comes to a hub page, he/she will find many useful links wich take him/her to good content pages on the topic.

HITS Algorithm

Authority and Hubs Pages

- If q it is good hub, then q point to many good authority pages p.
- If p it is a good authority, then p is pointed by many good hub pages q.
- Authorities and Hubs have a **mutual reinforcement** relationship.
- We can give a measure of the quality of *goodness* for authority and hubs. We call it: Authority and Hub weight (a_p, h_q)





HITS Algorithm

How it works

Assumptions

- The authority level (or rank) came from in-edges.
- A good authority come from good hubs and a good hub contains links that point to good authorities.

A **simple method** to differentiate the page's relevance is:

- First **assigning non-negative weights**, depending if the page is hub or authoritative. Well, finally the page have both of them.
- Next, the weights are **adjusted by an iterative process** and the **relative page's importance** in the community is calculated.

HITS Algorithm

How it works

Given a query q, HITS collects a set of pages as follows:

1. It send the query q to a search engine system and collects t highest ranked pages, which assume to be highly relevant to the search query. This is called the **root** set W.
2. It then grows W by including any page pointed to by a page in W and any page that points to a page in W. It restrict its size by allowing each page in W to bring at most k pages. This set is called the **base set** S.

HITS Algorithm

How it works

Assuming we have a set S of pages connected where:

- V is the set of pages (or nodes).
- E is the set of directed edges (or links).
- Then $G = (V, E)$
- We use L to denote the adjacency matrix of the graph, where:

$$L = \begin{cases} 1 & \text{if } \quad (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$

HITS Algorithm

How it works

- Let the authority score of the page i be $a(i)$, and the hub score of the page i be $h(i)$.
- The mutual reinforcing relationship of the two scores is represented as:

$$a(i) = \sum_{(j,i) \in E} h(j)$$

The authority score of the page i correspond to the sum of all the hubs that are point to it.

$$h(i) = \sum_{(i,j) \in E} a(j)$$

The hub score of the page i correspond to the sum of all the authorities it points to.

HITS Algorithm

How it works

- Then, we use ***a*** to denote the column vector with all the authority scores and ***h*** to denote the column vector with all the hub scores.
- This is equivalent to say:

$$a = L^T h$$

$$h = La$$

- If we use a_k and h_k to denote authority and hub scores at the k-th iteration, the iterative processes are:

$$a_k = L^T L a_{k-1} \iff a_{k+1} = L^T L a_k$$

$$h_k = L L^T h_{k-a} \iff h_{k+1} = L L^T h_k$$

HITS Algorithm

HITS Algorithm Steps

1. Initialize $a = (1, \ldots, 1)$ and $h = (1, \ldots, 1)$
2. Calculate:

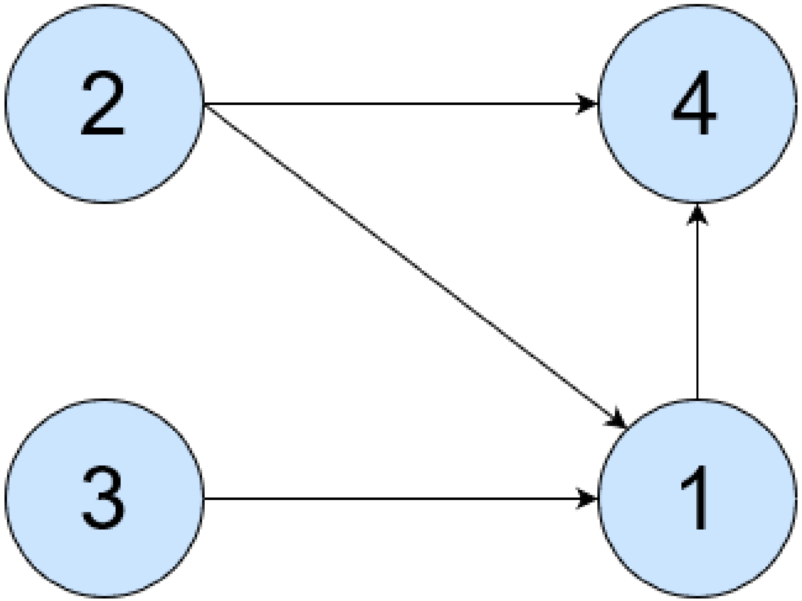
$$a_{k+1} = L^T L a_k$$

$$h_{k+1} = L L^T h_k$$

3. Normalize a_k and h_k
4. Repeat until: $\left\| a_{k+1} - a_k \right\|_1 < \varepsilon_a \wedge \left\| h_{k+1} - h_k \right\|_1 < \varepsilon_h$
5. Get the most auth pages and the most hub pages.

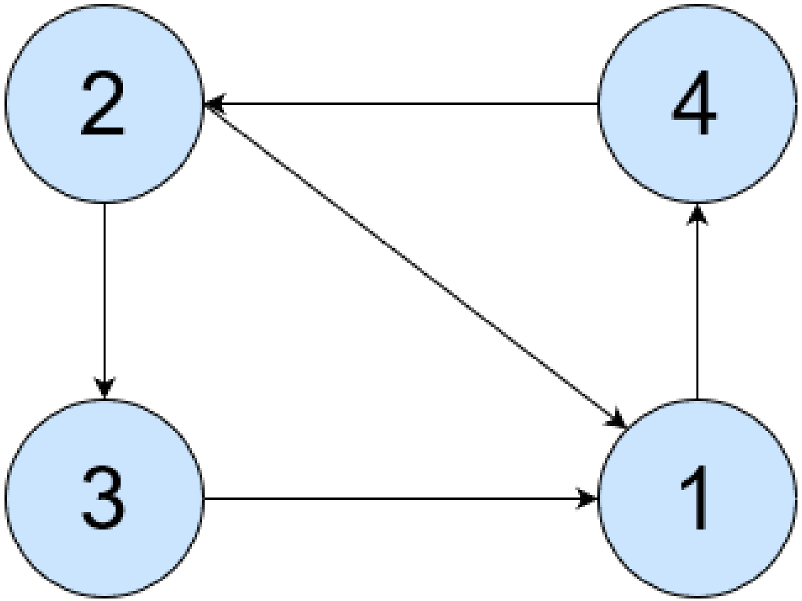
HITS Algorithm

Example



HITS Algorithm

Example



HITS Algorithm

How it works

By construction of HITS:

- After using the query to extract the pages related with the query's terms, the page's text content is ignored in the page's rank task.
- Being the algorithm purely **hyperlink-based computation**.

CLEVER Project (Chakrabarti S.)

- Addresses the problem by considering query's terms in the calculus of the above equations.
- A non-negative weight, whose initial value is basis on the text that surround the hyperlink expression (a tag in HTML)

HITS Algorithm

Advantages and Disadvantages

Advantages

- Double ranking (by Authority and by Hub).
- Rank pages according to a query topic.

Disadvantages

- Doesn't have the anti-spam capability of PageRank.
- Topic drift.
- Query-dependence.
- Query time evaluation.

Outline

Page Rank

HITS

SALSA

SALSA

What is it?

Source

"Google's PageRank and Beyond: The Science of Search Engine Rankings". Amy N. Langville and Carl D. Meyer

- Proposed by R. Lempel and S. Moran in 2001.
- SALSA stands for Stochastic Approach to Link Structure Analysis.
- Like HITS, SALSA create both Authority and Hub scores for webpages.
- SALSA creates a neighborhood graph showing the closeness between Authority pages and Hub pages.

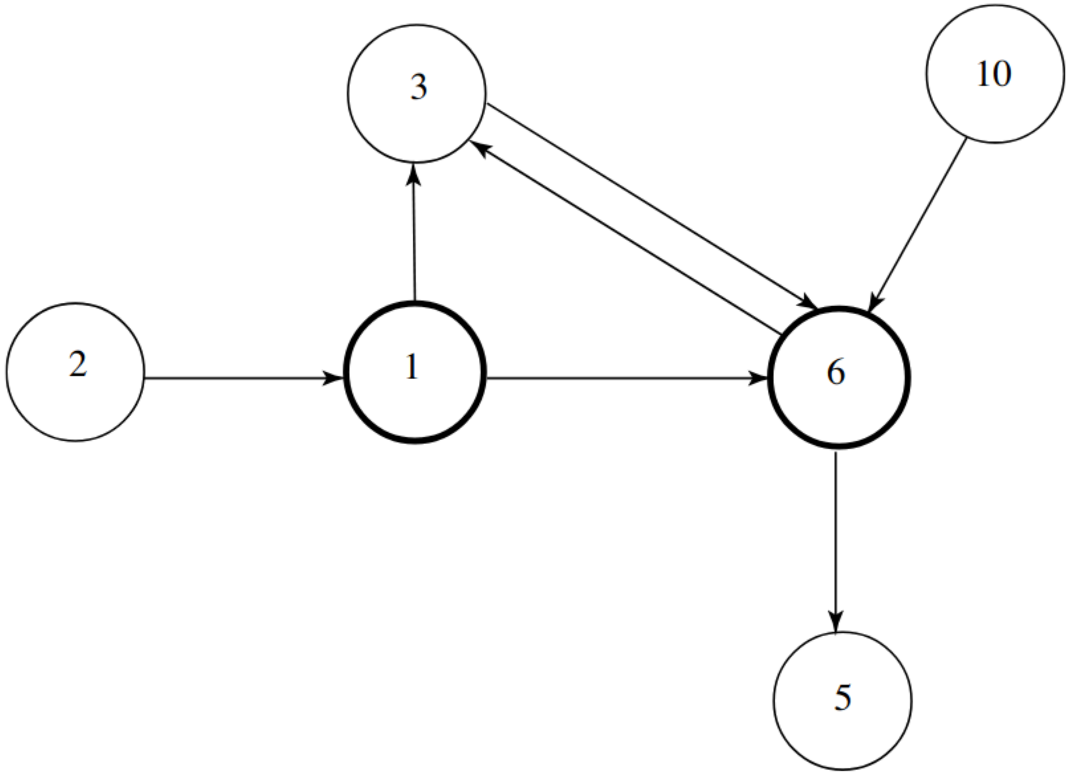
Other Algorithms

SALSA

- Rather than forming and adjacency matrix L for the neighborhood graph N, a bipartite unidirected graph, denoted G, is built.
- G is defined by the sets:
  - V_h: Set of Hub nodes (all nodes in N with outdegree > 0).
  - V_a: Set of Authority nodes (all nodes in N with indegree > 0).
  - E: Set of directed edges in N.
- Note that a node in N may be in both V_h and V_a.

Other Algorithms

SALSA Example



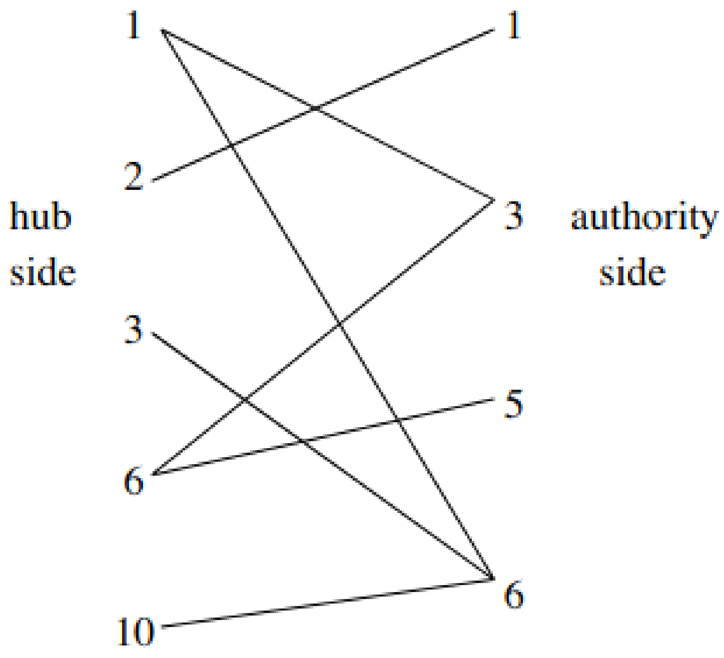
Other Algorithms

SALSA Example

- From the previous graph, we have that:

$$V_h = \{1, 2, 3, 6, 10\} \qquad V_a = \{1, 3, 5, 6\}$$

- Then, we can create the bipartite undirected graph G, as shown.
- Every directed edge in E is represented by an uni-directedted edge in G.



Other Algorithms

Where does SALSA fit in?

- Matrices H and A can be derived from the adjacency matrix L used in the HITS and PageRank methods.
- HITS used unweighted matrix L.
- PageRank uses a row weighted version of matrix L.
- SALSA uses both row and column weighting.

Other Algorithms

How are A and H computed in SALSA?

To get A and H matrices, we have to calculate normalize versions of L:

- Let L_r be L with each nonzero row divided by its row sum.
- Let L_c be L with each nonzero column divided by its column sum.

In the previous example, it will be:

$$L = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array} \quad L_r = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array} \quad L_c = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 0 & 0 & \frac{1}{2} & 0 & \frac{1}{3} & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{3} & 0 \end{pmatrix} \end{array}$$

Other Algorithms

How are A and H computed in SALSA?

- H: SALSA's hub matrix, consists of the nonzero rows and columns of $L_r L_c^T$
- A: SALSA's authority matrix, consists of the nonzero rows and columns of $L_c^T L_r$

$$L_r L_c^T = \begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} & \begin{pmatrix} \frac{5}{12} & 0 & \frac{1}{6} & \frac{1}{2} & \frac{1}{6} & \frac{1}{6} \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & 0 & \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \end{pmatrix} \end{array}$$

$$L_c^T L_r = \begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} & \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{6} & 0 & \frac{5}{6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Other Algorithms

How are A and H computed in SALSA?

$$H = \begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 6 & 10 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 6 \\ 10 \end{array} & \begin{pmatrix} \frac{5}{12} & 0 & \frac{1}{6} & \frac{1}{6} & \frac{2}{12} \\ 0 & 1 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ \frac{1}{4} & 0 & 0 & \frac{3}{4} & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \end{pmatrix} \end{array}$$

$$A = \begin{array}{c c} & \begin{array}{cccc} 1 & 3 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 3 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{6} & 0 & \frac{5}{6} \end{pmatrix} \end{array}$$

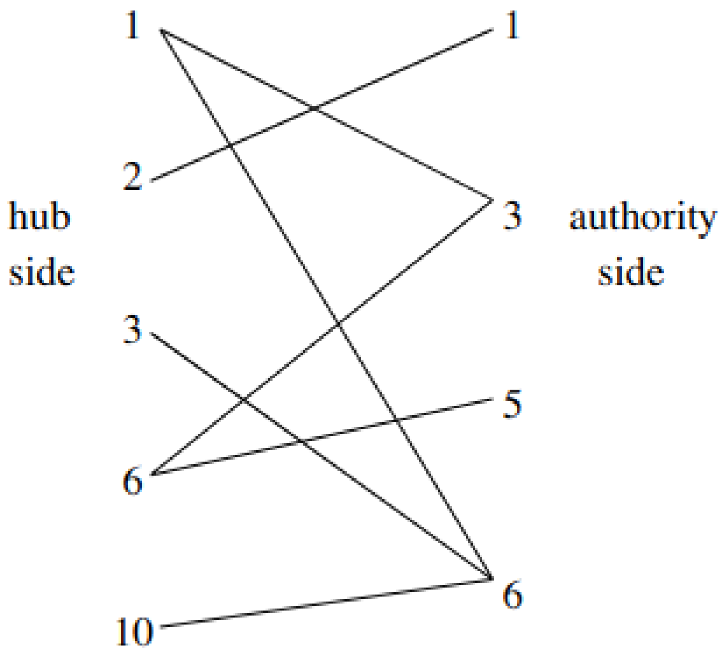
Other Algorithms

The Power Method

- If the bipartite graph G is *connected*, then A and H are both irreductible Markov chains and π_h^T, the stationary vector of H, gives the hub scores for the query with neighborhood graph N, and π_a^T gives the authority scores.
- If G is not *connected*, then A and H contain multiple irreducible components. In this case, the global authority and hub scores must be pasted together from the stationary vectors for each individual irreducible component.

Other Algorithms

The Power Method Example



- Because G is **not connected**, A and H contain multiple connected components.
- H contains two connected components:

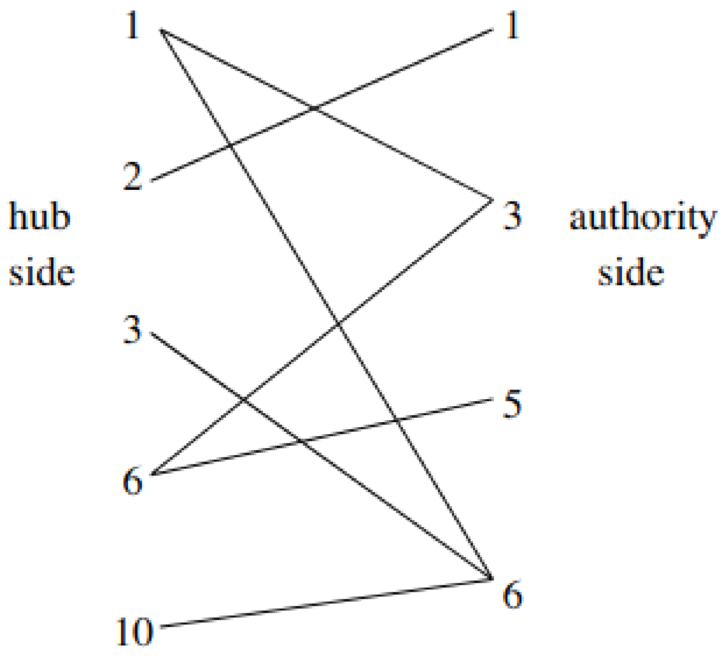
$$C = \{2\} \qquad D = \{1, 3, 6, 10\}$$

- A contains two connected components:

$$E = \{1\} \qquad F = \{3, 5, 6\}$$

Other Algorithms

The Power Method Example



- The stationary vectors for the two irreducible components of H are:

$$\pi_h^{T(C)} = \begin{pmatrix} 1 \end{pmatrix} \qquad \pi_h^{T(D)} = \begin{pmatrix} \frac{1}{3} & \frac{1}{6} & \frac{1}{3} & \frac{1}{6} \end{pmatrix}$$

- The stationary vectors for the two irreducible components of H are:

$$\pi_a^{T(E)} = \begin{pmatrix} 1 \end{pmatrix} \qquad \pi_a^{T(F)} = \begin{pmatrix} \frac{1}{3} & \frac{1}{6} & \frac{1}{2} \end{pmatrix}$$

Other Algorithms

The Power Method Example

- Now, we can join the two components together for each matrix.
- We must multiply each entry in the vector by its appropiate weight.

Other Algorithms

The Power Method Example

- Thus the global Hub vector is:

$$\pi_h^T = \left(\frac{4}{5} \times \frac{1}{3} \;\; \frac{1}{5} \times 1 \;\; \frac{4}{5} \times \frac{1}{6} \;\; \frac{4}{5} \times \frac{1}{3} \;\; \frac{4}{5} \times \frac{1}{6}\right)$$

$$\pi_h^T = (0.2667 \;\; 0.2 \;\; 0.1333 \;\; 0.2667 \;\; 0.1333)$$

Other Algorithms

The Power Method Example

- And the global Authority vector is:

$$\pi_a^T = \begin{pmatrix} \frac{1}{4} \times 1 & \frac{3}{4} \times \frac{1}{3} & \frac{3}{4} \times \frac{1}{6} & \frac{3}{4} \times \frac{1}{2} \end{pmatrix}$$

$$\pi_a^T = (0.25 \;\; 0.25 \;\; 0.125 \;\; 0.375)$$

Other Algorithms

Advantages and Disadvantages of SALSA

Advantages

- Not affected as much my topic drift like HITS.
- Less affected susceptible to spamming.
- Dual rank (Authority and Hubs).

Disadvantages

- Query-dependence.
- Query time evaluation.

Artificial Intelligence Applied to the Web

Chapter 3 Part 2 - Web Page Rank Algorithms

Diego Cornejo, Felipe Hernández and Juan Velásquez

University of Chile
Departament of Industrial Engineering

Spring 2025