Link Analysis I

Alexander Schönhuth

UNIVERSITÄT BIELEFELD
Faculty of Technology

Bielefeld University
June 2, 2022

TODAY

Overview

- PageRank: Introduction, Definition
- PageRank Reality: Structure of the Web
- Topic-Sensitive PageRank: Classify Pages by Topics

Learning Goals: Understand these topics and get familiarized

PageRank
Introduction

PageRank: Overview

- Motivation of PageRank definition: history of search engines
- Concept of *random surfers* foundation of PageRank's effectiveness
- *Taxation* ("recycling of random surfers") allows to deal with problematic web structures

History: Early Search Engines

- *Early search engines*
  - Crawl the (entire) web
  - List all terms encountered in an *inverted index*
  - An inverted index is a data structure that, given a term, provides pointers to all places where term occurs
- On a *search query* (a list of terms)
  - pages with those terms are extracted from the index
  - ranked according to use of terms within pages
  - E.g. the term appearing in the header renders page more important
  - or the term appearing very often

Term Spam

- *Spammers* exploited this to their advantage
- *Simple strategy:*
  - Add terms thousands of times to own webpages
  - Terms can be made hidden by using background color
  - So pages are listed in searches that do not relate to page contents
  - Example: add term "movie" 1000 times to page that advertizes shirts
- *Alternative strategy:*
  - Carry out web search on term
  - Copy-paste highest ranked page into own page
  - Upon new search on term, own page will be listed high up
- Corresponding techniques are referred to as *term spam*

PageRank's Motivation: Fighting Term Spam

Idea:

- Simulate *random web surfers*
  - They start at random pages
  - They randomly follow web links leaving the page
  - Iterate this procedure sufficiently many times
  - Eventually, they gather at "important" pages
- Judge page also by *contents of surrounding pages*
  - Difficult to add terms to pages not owned by spammer

PageRank's Motivation: Fighting Term Spam

Justification

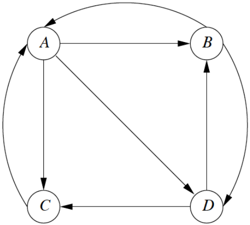
- Ranking web pages by number of in-links does not work
  - Spammers create "spam farms" of dummy pages all linking to one page
- *But*, spammers' pages do not have in-links from elsewhere
- ☞ Random surfers do not wind up at spammers' pages
- (Non-spammer) page owners place links to pages they find helpful
- Random surfers indicate which pages are likely to visit
  ☞ Users are more likely to visit useful pages

PageRank: Definition

- PageRank is a function that assigns a real number to each (accessible) web page
- *Intuition:* The higher the PageRank, the more important the page
- There is not one fixed algorithm for computing PageRank
- There are many variations, each of which caters to particular issue

PageRank: Definition

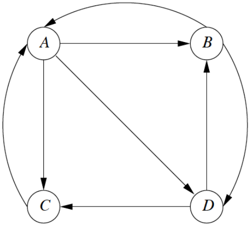
- Consider the web as a directed graph
  - Nodes are web pages
  - Directed edges are links leaving from and leading to web pages



Hypothetical web with four pages

Adopted from `mmds.org`

PageRank: Definition

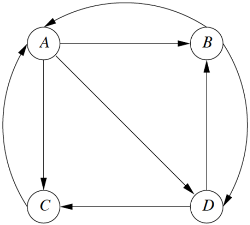


Random walking a web with four pages

Adopted from `mmds.org`

- For example, a *random surfer* starts at node A
- Walks to B, C, D each with probability $1/3$
- So has probability 0 to be at A after first step

PageRank: Definition



Random walking a web with four pages

Adopted from `mmds.org`

- *Random surfer* at B, for example, in next step
  - is at A, D each with probability $1/2$
  - is at B, C with probability 0

Web Transition Matrix: Definition

Definition [Web Transition Matrix]:

- Let n be the number of pages in the web
- The *transition matrix* $M = (m_{ij})_{1 \le i,j \le n} \in \mathbb{R}^{n \times n}$ has n rows and columns
- For each $(i, j) \in \{1, ..., n\} \times \{1, ..., n\}$
  - $m_{ij} = 1/k$, if page j has k arcs out, of which one leads to page i
  - $m_{ij} = 0$ otherwise

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

Transition matrix for web from slides before

Adopted from `mmds.org`

PageRank Function: Definition

Definition [PageRank Function]:

- Let n be the number of pages in the web
- Let $p_i^t, i = 1, ..., n$ be the probability that the random surfer is at page i after t steps
- The *PageRank function* for $t \geq 0$ is defined to be the vector

$$p^t = (p_1^t, p_2^t, ..., p_n^t) \in [0, 1]^n$$

PageRank Function: Interpretation

- Usually, $p^0 = (1/n, ...1/n)$ for each $i = 1, ..., n$
- So before the first iteration, the random surfer is at each page with equal probability
- The probability to be at page i in step $t+1$ is the sum of probabilities to be at page j in step t times the probability to move from page j to i
- That is, $p_i^{t+1} = \sum_{j=1}^{n} m_{ij} p_j^t$ for all i, t, or, in other words

$$p^{t+1} = Mp^t \quad \text{for all } t \geq 0 \tag{1}$$

- So, applying the web transition matrix to a PageRank function yields another one

PAGERANK FUNCTION: MARKOV PROCESSES

$$\boxed{p^{t+1} = Mp^t \quad \text{for all } t \geq 0}$$

- This relates to the theory of *Markov processes*
- Given that the web graph is *strongly connected*
  - That is: one can reach any node from any other node
  - In particular, there are no *dead ends*, nodes with no arcs out
- it is known that the surfer reaches a *limiting distribution* $\bar{p}$, characterized by

$$M\bar{p} = \bar{p} \tag{2}$$

PageRank Function: Markov Processes

$$\boxed{M\bar{p} = \bar{p}}$$

- Further, because M is *stochastic* (= columns each add up to one)
  - $\bar{p}$ is the *principal eigenvector*, which is
  - the eigenvector associated with the largest eigenvalue, which is one
- $\bar{p}_i$ is the probability that the surfer is at page i after a long time
- Principal eigenvector of M expresses where the surfer will end up
- *Reasoning:* The greater $\bar{p}_i$, the more important page i

p_i^{-} is page rank of page i

PageRank Function: Computation

$$\boxed{M\bar{p} = \bar{p}}$$

- It holds that

$$M^t p^0 \quad \underset{t\to\infty}{\longrightarrow} \quad \bar{p} \tag{3}$$

p^0 -> Mp^0 = p^1 -> Mp^1 = p^2 -> …
p^{-}

- So, for *computing* $\bar{p}$, apply iterative matrix-vector multiplication until (approximate) convergence
- *Example:* Iterative application of transition matrix from above

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 9/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix}, \begin{bmatrix} 15/48 \\ 11/48 \\ 11/48 \\ 11/48 \end{bmatrix}, \begin{bmatrix} 11/32 \\ 7/32 \\ 7/32 \\ 7/32 \end{bmatrix}, \dots, \begin{bmatrix} 3/9 \\ 2/9 \\ 2/9 \\ 2/9 \end{bmatrix}$$

Convergence to limiting distribution for four-node web graph

Adopted from `mmds.org`

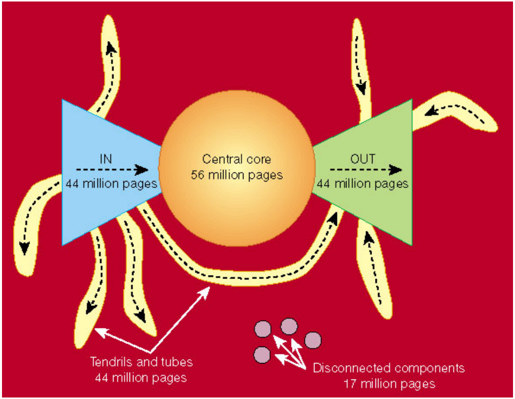
PageRank Function: Computation

$$\boxed{M\bar{p} = \bar{p}}$$

- It holds that
$$M^t p_0 \quad \underset{t \to \infty}{\longrightarrow} \quad \bar{p} \tag{4}$$
- So, for *computing* $\bar{p}$, apply iterative matrix-vector multiplication until (approximate) convergence
- In practice, working real web graphs
  - 50-75 iterations do just fine
  - For *efficient computation*, recall MapReduce based matrix-vector multiplication techniques

PageRank Reality
Dead Ends and Spider Traps

STRUCTURE OF THE WEB



Bowtie picture of the web

Adopted from `mmds.org`

Web Bowtie: Summary

- *Strongly connected component (SCC):* core of the web
- *In-component (IC):*
  - One can reach SCC from IC
  - but not return to IC once left
- *Out-component (OC):*
  - Can be reached from SCC
  - but no longer be left
- *Tendrils:*
  - *First type:* reachable from IC, but can no longer be left
  - *Second type:* can reach OC, but cannot be returned to
- *Tubes:*
  - Can be reached from IC
  - Can only reach OC
- *Isolated components* are not reachable from and cannot reach other components

Bowtie and Markov Chains

Issue: Limiting Distribution

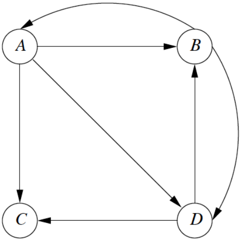
- Random surfers will inevitably wind up in out-component
- Limiting distribution has probability 0 on IC and SCC

☞ No page in IC or SCC of importance

PageRank Modification

- Avoid *dead ends*, single pages with no outlinks
- Avoid *spider traps*, sets of pages without dead ends, but no arcs out
- *Solution: Taxation*
  - Assume random surfer has small probability to leave the web
  - Instead, new surfer starts at random node of the web

Dead Ends



Web graph with dead end (node C)

Adopted from `mmds.org`

- Dead end = columns of all zeroes in the web transition matrix M
- M then is *substochastic* (= column sums at most 1)
- $M^i v$ yields vector with zeroes for certain components
- Dead ends *drain out* the web

Dead Ends

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

Transition matrix for web with dead end (node C)

Adopted from `mmds.org`

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 3/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix}, \begin{bmatrix} 5/48 \\ 7/48 \\ 7/48 \\ 7/48 \end{bmatrix}, \begin{bmatrix} 21/288 \\ 31/288 \\ 31/288 \\ 31/288 \end{bmatrix}, \dots, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Corresponding limiting distribution

Adopted from `mmds.org`

AVOIDING DEAD ENDS

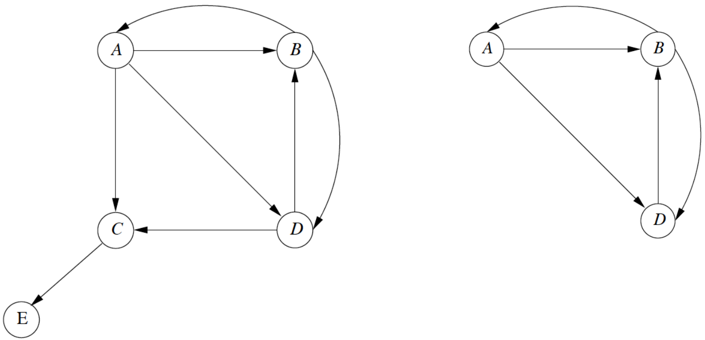
Dropping dead ends: Procedure

- Drop dead ends from graph, and corresponding edges
- Dropping dead ends may create more dead ends
- Keep dropping dead ends iteratively

Dropping dead ends: Consequences

- Removes parts of out-component, tendrils and tubes
- Leaves SCC and in-component

Avoiding Dead Ends



Graph before (left) and after iterative removal of dead ends (right)

DROPPING DEAD ENDS: PAGERANK COMPUTATION

1. After iterative removal of dead ends, compute PageRank for remaining core nodes
2. Re-introduce nodes iteratively, in reverse order relative to their removal
3. PageRank for re-introduced node: sum over all predecessors, PageRank of predecessor p divided by the number of successors of p

Dead Ends

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

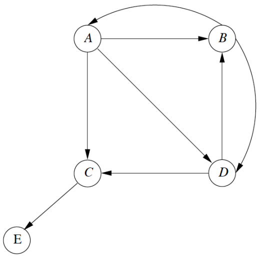
Transition matrix after removal of dead ends

$$\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}, \begin{bmatrix} 1/6 \\ 3/6 \\ 2/6 \end{bmatrix}, \begin{bmatrix} 3/12 \\ 5/12 \\ 4/12 \end{bmatrix}, \begin{bmatrix} 5/24 \\ 11/24 \\ 8/24 \end{bmatrix}, \ldots, \begin{bmatrix} 2/9 \\ 4/9 \\ 3/9 \end{bmatrix}$$

PageRank(A) = 2/9, PageRank(B) = 4/9, PageRank(D) = 3/9

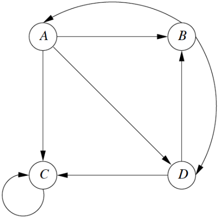
Adopted from `mmds.org`

Dead Ends: PageRank Computation



1. From core: PageRank(A) = 2/9, PageRank(B) = 4/9, PageRank(D) = 3/9
2. Re-introduce node C first:
   $\text{PageRank}(C) = 1/3 \times \text{PageRank}(A) + 1/2 \times \text{PageRank}(D) = \frac{13}{54}$
3. Then re-introduce node E: $\text{PageRank}(E) = 1 \times \text{PageRank}(C) = \frac{13}{54}$

SPIDER TRAPS



Web graph with spider trap (set containing single node C)

Adopted from `mmds.org`

- (Small) group of nodes with no dead ends, but no arcs out
- Can appear intentionally or unintentionally
- “Soak up” all PageRank

Spider Traps

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

Transition matrix for web with single node spider trap (third column)

Adopted from `mmds.org`

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 3/24 \\ 5/24 \\ 11/24 \\ 5/24 \end{bmatrix}, \begin{bmatrix} 5/48 \\ 7/48 \\ 29/48 \\ 7/48 \end{bmatrix}, \begin{bmatrix} 21/288 \\ 31/288 \\ 205/288 \\ 31/288 \end{bmatrix}, \ldots, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Corresponding limiting distribution

Adopted from `mmds.org`

Spider Traps: Taxation

- Allow the random surfer to get *teleported* to a random page
- *Notation:*
  - Let n be the total number of web pages
  - Let $\mathbf{e} := (1, ..., 1)$ be the vector of length n with all entries one
  - Let β be a small constant; usually $0.8 \leq \beta \leq 0.9$
- *Taxation:* In each matrix-vector multiplication iteration, instead of just computing $\mathbf{v}' = M\mathbf{v}$, compute

$$\mathbf{v}' = \beta M\mathbf{v} + \frac{1}{n}(1-\beta)\mathbf{e} = \beta M\mathbf{v} + (1-\beta)(\frac{1}{n}, ..., \frac{1}{n})^T \qquad (5)$$

to obtain a new vector $\mathbf{v}'$ from the actual one $\mathbf{v}$

Spider Traps: Taxation

- *Taxation:* In each matrix-vector multiplication iteration, instead of just computing $\mathbf{v}' = M\mathbf{v}$, compute

$$\mathbf{v}' = \beta M\mathbf{v} + (1-\beta)(\frac{1}{n}, ..., \frac{1}{n})^T$$

  to obtain a new vector $\mathbf{v}'$ from the actual one $\mathbf{v}$

- *Interpretation:*
  - With probability β, the surfer follows an out-link
  - With probability $1-\beta$, the surfer get teleported to a random page
  - In dead ends, surfer disappears with probability β
  - So if there are dead ends, sum of entries in v' less than one
    ☞ So remove dead ends first

Spider Traps

$$\mathbf{v}' = \begin{bmatrix} 0 & 2/5 & 0 & 0 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 0 & 4/5 & 2/5 \\ 4/15 & 2/5 & 0 & 0 \end{bmatrix} \mathbf{v} + \begin{bmatrix} 1/20 \\ 1/20 \\ 1/20 \\ 1/20 \end{bmatrix}$$

Iteration with taxation, with spider trap (third column)

Adopted from `mmds.org`

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 9/60 \\ 13/60 \\ 25/60 \\ 13/60 \end{bmatrix}, \begin{bmatrix} 41/300 \\ 53/300 \\ 153/300 \\ 53/300 \end{bmatrix}, \begin{bmatrix} 543/4500 \\ 707/4500 \\ 2543/4500 \\ 707/4500 \end{bmatrix}, \ldots, \begin{bmatrix} 15/148 \\ 19/148 \\ 95/148 \\ 19/148 \end{bmatrix}$$

Corresponding limiting distribution

Adopted from `mmds.org`

PageRank: Efficient Computation

- PageRank virtually is matrix-vector multiplication
  - Consider MapReduce techniques (originally motivated by PageRank)
- *Caveats*, however:
  - Transition matrix M is very sparse; consider appropriate representation of M
  - To reduce communication cost, use combiners
  - Earlier striping technique not sufficient
- So, additional techniques necessary:

see `https://mmds.org`, section 5.2

Topic-Sensitive PageRank

Topic-Sensitive PageRank: Motivation

- Different people have different interests, but ...
- ... different interests are expressed by identical terms
  - E.g. `jaguar` may refer to animal, automobile, operating system, game console
- *Ideally:* Each user has private PageRank vector that measures individual importance of pages
- *But:* It is not feasible to store a vector of length many billions for one billion users

Topic-Sensitive PageRank: Basic Idea

- Identify a (rather small) number of topics
- Compute topic specific PageRank vectors
  - Store topic vectors ...
  - ... instead of individual user vectors
  - There are much less topic vectors
  - *Example for useful topics:* See `https://www.curlie.org/` (new) or `https://www.dmoz-odp.org` for top-level categories
- Assign users to (weighted combination of) topic vectors
- *Drawback:* Looses accuracy
- *Benefit:* Saves massive amounts of space

Topic-Sensitive PageRank: Computation

Idea: Biased Random Walks

- Simulate random surfers that are to prefer pages adhering to particular topics
- Random surfers start at approved topic-specific pages only
- When surfing, they will preferably visit pages linked from topic-specific pages
- Such pages are likely to deal with topic as well
- When being re-introduced (to avoid dead ends, spider traps), surfers again start at approved pages

Topic-Sensitive PageRank: Definition

- Let S be the *teleport set*, i.e. the pages that are approvedly topic-specific
- Let $n, \mathbf{v}, \mathbf{v}', M, \beta$ be as before
- Let $\mathbf{e}_S \in \{0,1\}^n$ be a bit vector of length n such that

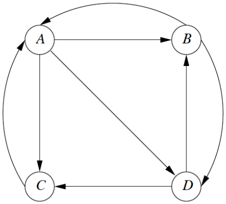
$$\mathbf{e}_S[i] = \begin{cases} 1 & \text{if } i\text{-th page belongs to } S \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Definition [Topic-Sensitive PageRank]
The *topic-sensitive PageRank for S* is the limit of the iteration

$$\mathbf{v}' = \beta M \mathbf{v} + (1-\beta)\frac{\mathbf{e}_S}{|S|} \tag{7}$$

where $|S|$ is the cardinality (size) of S.

Topic-Sensitive PageRank: Example



Example web graph

Adopted from `mmds.org`

$$\beta M = \begin{bmatrix} 0 & 2/5 & 4/5 & 0 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 2/5 & 0 & 0 \end{bmatrix}$$

Corresponding weighted web transition matrix

Adopted from `mmds.org`

Topic-Sensitive PageRank: Example II

$$\mathbf{v}' = \begin{bmatrix} 0 & 2/5 & 4/5 & 0 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 2/5 & 0 & 0 \end{bmatrix} \mathbf{v} + \begin{bmatrix} 0 \\ 1/10 \\ 0 \\ 1/10 \end{bmatrix}$$

Topic sensitive PageRank computation iteration for teleport set {B,D}

Adopted from `mmds.org`

$$\begin{bmatrix} 0/2 \\ 1/2 \\ 0/2 \\ 1/2 \end{bmatrix}, \begin{bmatrix} 2/10 \\ 3/10 \\ 2/10 \\ 3/10 \end{bmatrix}, \begin{bmatrix} 42/150 \\ 41/150 \\ 26/150 \\ 41/150 \end{bmatrix}, \begin{bmatrix} 62/250 \\ 71/250 \\ 46/250 \\ 71/250 \end{bmatrix}, \ldots, \begin{bmatrix} 54/210 \\ 59/210 \\ 38/210 \\ 59/210 \end{bmatrix}$$

Corresponding limiting distribution

Adopted from `mmds.org`

Topic-Sensitive PageRank: Practical Considerations

- Pick an appropriate set of topics
- For each topic selected, determine teleport set
- *Classifying documents by topic*
  - Has been studied in great detail
  - Topics are characterized by words relating to topic
  - Such words appear surprisingly often in topic-specific pages
  - Determine such words from pages known to relate to topic beforehand
  - Remember the TF.IDF measure (first lecture)

Topic-Sensitive PageRank: Practical Considerations

- When confronted with search query, decide on related topics
- *Determining user-specific topics:*
  - Allow user to choose from menu
  - Infer topics from words appearing in recent queries
  - Infer topics from information on user (bookmarks, stated interests in social media,...)
- Use corresponding topic-sensitive PageRank vectors for ranking responses

MATERIALS / OUTLOOK

- See *Mining of Massive Datasets*, chapters 5.1; 5.3 – 5.5
- As usual, see `http://www.mmds.org/` in general for further resources
- Next lecture: “Frequent Itemsets I”
  - See *Mining of Massive Datasets* chapter 6.1, 6.2, 6.3.1, 6.4.1, 6.4.2