Mathematics for Machine Learning (AI 512):

Amit Chattopadhyay

IIIT-Bangalore

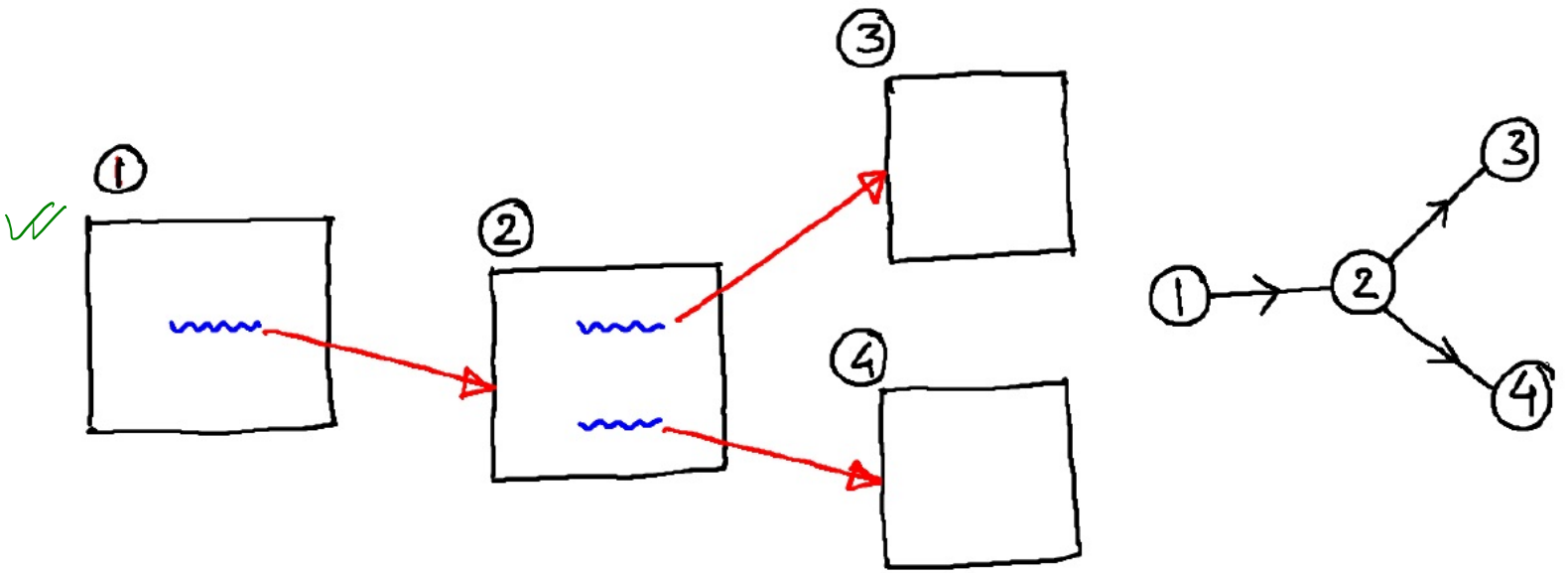
Lecture: Page Rank Algorithm

Page Rank Algorithm

WWW: As a Directed Graph

- **World Wide Web (www):** considered as a **directed graph**,
  node/vertex: corresponds to a page,
  edge: corresponds to a hyperlink
  (e.g. Wikipedia, Citations, Encyclopedia etc.)

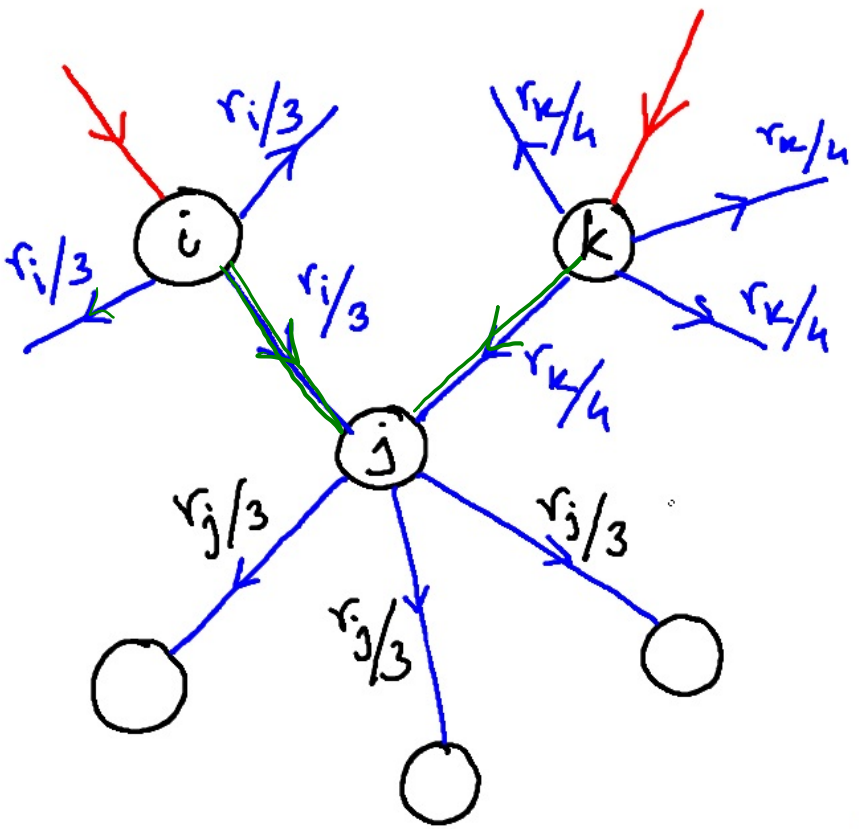
- **Goal:** To establish the importance (**rank**) of pages on www



Page Rank: The "Flow" Model

1. A page is more important if it has more **in-links**
2. Links from important pages count more
   (i.e. a vote from an important page worth more)

Ex.



- If page i with rank r_i has c_i out-links, each out-link gets $\frac{r_i}{c_i}$ votes
- Page j's own rank r_j is the sum of the votes on its in-links

$$r_j = \frac{r_i}{3} + \frac{r_k}{4}$$

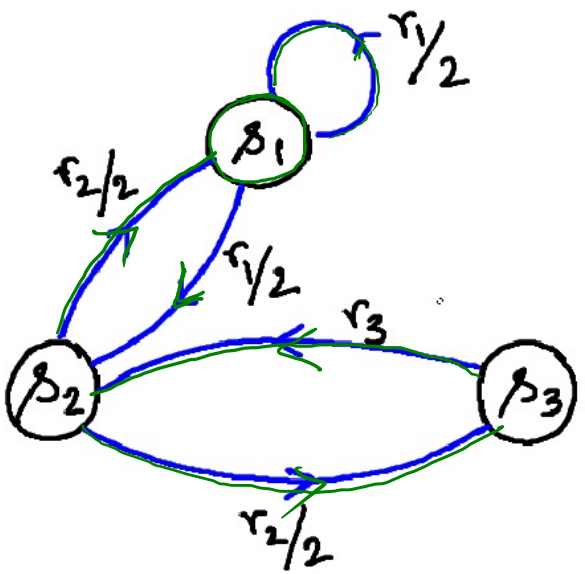
Page Rank: The "Flow" Model

Define: Rank r_j of a node j

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

the summation is taken over all nodes i point to j and d_i is the out-degree of i.

Example 1:

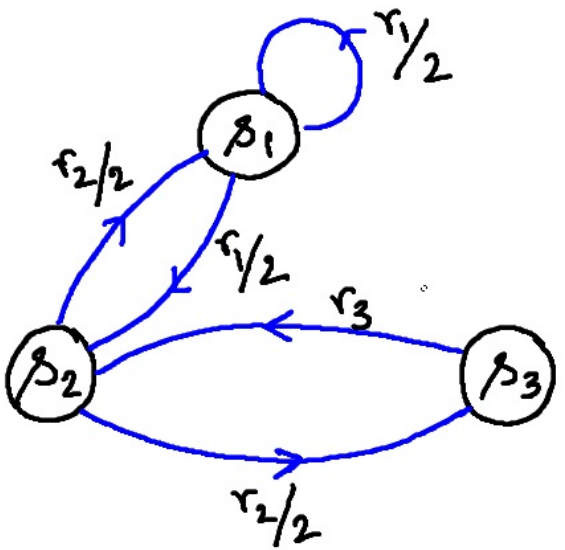


Flow equations:

$$r_1 = \frac{r_1}{2} + \frac{r_2}{2}$$
$$r_2 = \frac{r_1}{2} + r_3$$
$$r_3 = \frac{r_2}{2}$$

Page Rank: The "Flow" Model

Ex. (Contd..)



Flow equations:

$$r_1 = \frac{r_1}{2} + \frac{r_2}{2}$$
$$r_2 = \frac{r_1}{2} + r_3$$
$$r_3 = \frac{r_2}{2}$$

$$\Leftrightarrow \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 1 & 0 \end{bmatrix}$$

$$\Leftrightarrow \underline{r} = \underline{r}\,\mathbb{M}$$

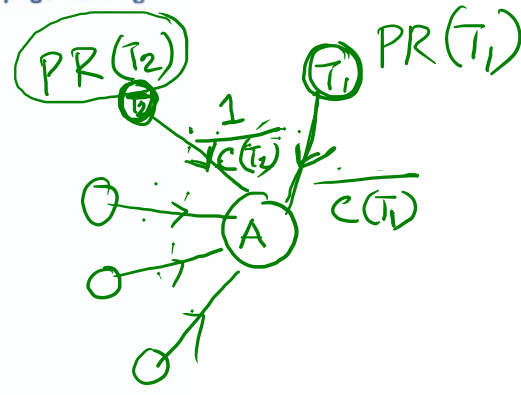
Solve: by Gaussian elimination? **not scalable**

Google Page Rank: Introduction

- by **Sergey Brin** and **Lawrence Page**, 1998, PhD students at CS, Standford University
- **Founders of Google**. Outlined the basics of **Page Rank Algorithm** used in **Google Search Engine**
- **Published in:** Computer Networks and ISDN Systems (30:107-117:1998)

Idea: Google Page Rank

Given below is the snapshot of the **original paper (in Computer Networks and ISDN Systems, 30: 107–117, 1998) of Sergey Brin and Lawrence (Larry) Page, Founders of Google,** where they outlined (among other things) the basics of the page-ranking algorithm they used in Google.



The Anatomy of a Large-Scale Hypertextual Web Search Engine

Sergey Brin and Lawrence Page

Computer Science Department,
Stanford University, Stanford, CA 94305, USA
sergey@cs.stanford.edu and page@cs.stanford.edu

Abstract

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/

This algorithm is outlined in this paper in a seemingly simple manner as below (excerpt reproduced from the above paper):

PageRank is defined as follows:

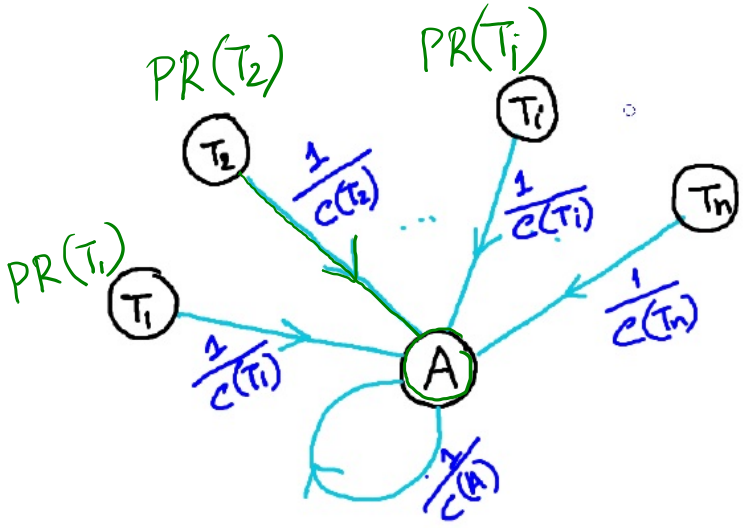
We assume page A has pages T1...Tn which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. We usually set d to 0.85. There are more details about d in the next section. Also C(A) is defined as the number of links going out of page A. The PageRank of a page A is given as follows:

$$PR(A) = (1-d) + d\,(PR(T1)/C(T1) + \ldots + PR(Tn)/C(Tn))$$

(check!) $\sum_{a \in A} PR(A) = N \neq 1.$

Note that the PageRanks form a probability distribution over web pages, so the sum of all web pages' PageRanks will be one.

Google Page Rank



$$PR(A) = PR(T_1)\,PR(A|T_1) + PR(T_2)\,PR(A|T_2) + \cdots + PR(T_n)\,PR(A|T_n)$$

$$= \frac{PR(T_1)}{C(T_1)} + \frac{PR(T_2)}{C(T_2)} + \cdots + \frac{PR(T_n)}{C(T_n)}$$

$t \backslash t+1$		T_1	T_2	A	T_n
$PR(T_1)$	T_1	$\frac{1}{C(T_1)}$	0	$\frac{1}{C(T_1)}$	$\cdots$
$PR(T_2)$	T_2	$\frac{1}{C(T_2)}$	$\frac{1}{C(T_2)}$	$\cdots$	$\frac{1}{C(T_2)}$
$\vdots$	$\vdots$				
$PR(T_n)$	T_n				

$$P_j^{(t+1)} = \sum_{i=1}^{n} P_i^{(t)} P_{i,j}$$

$$PR(A) = PR(T_1)\,PR(A|T_1) + \cdots$$

Page Rank : using Random Walk

- In a directed graph: **stationary probability** of that vertex
- In www: **Amount of time** or **frequency ratio** with which a page (vertex) will be visited over a long period of time

Page Rank: Random Walk Model

Outline: Random surfer on the web:

1. at any time t, surfer is on page i
2. at time $t+1$, the surfer selects an out-link uniformly at random and ends up on some page j linked from i
3. the process is repeated indefinitely

Page Ranks

- $\underline{p}^{(t)}$: **row vector** whose i-th coordinate is the probability that the surfer is at page i at time t
- **Finally, rank pages** according as their **stationary probabilities**

Random Walk Model: Transition Matrix

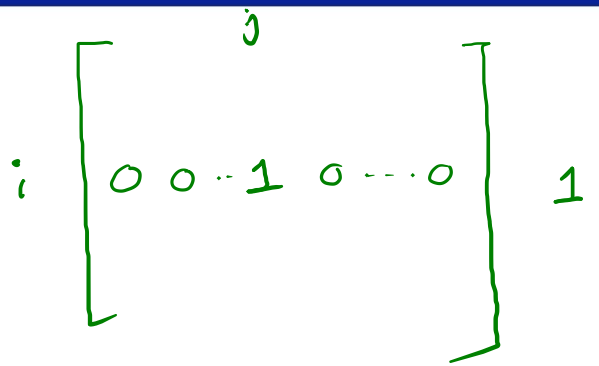
$$p_{i,j} = \begin{cases} \frac{1}{C(i)} & \text{if page } v_i \text{ points to page } v_j \\ 0 & \text{otherwise} \end{cases}$$

where $C(i)$ is the **out-degree** of vertex i

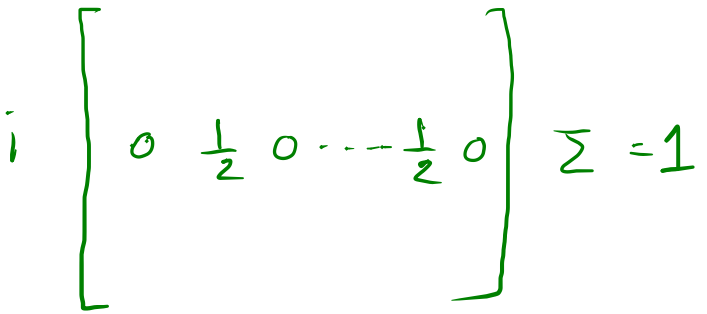
- $p_{i,j}$: transition probability of the RW, at a vertex v_i at time t, to select an edge (v_i, v_j) for moving to a vertex v_j at $t+1$
- $\mathbb{P} = (p_{i,j})_{N \times N}$

Random Walk Model: Transition Matrix

$C(i) = 1$

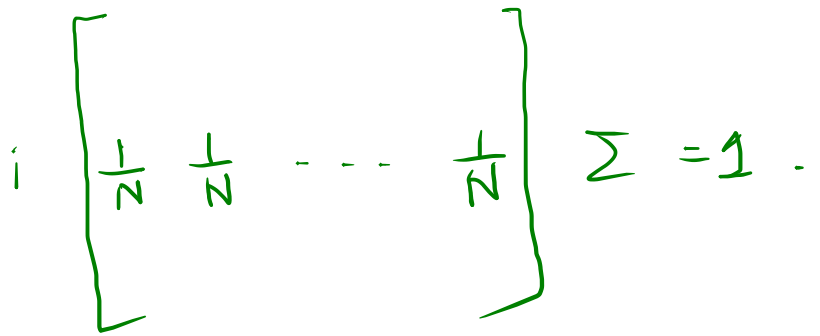


$C(i) = 2$



$\vdots$

$C(i) = N$



The Stationary Distribution

Q: Where is the surfer at time $t+1$?

- $\underline{p}^{(t+1)} = \underline{p}^{(t)} . \mathbb{P}$
- After reaching a **stationary distribution**: $\underline{\pi} = \underline{\pi}\mathbb{P}$
- $\underline{\pi}$ is also the left eigenvector of $\mathbb{P}$ corresponding to its largest eigenvalue $\lambda = 1$

Solving for '$\underline{\pi}$': Power Iteration Method

PageRank := Stationary distribution of the Random Walk

Power Iterative Method (assuming Convergence)

Input: A web graph with N nodes

Output: Ranks of nodes

1. **Initialize:** $\underline{p}^{(0)} = \begin{bmatrix} \frac{1}{N} & \dots & \frac{1}{N} \end{bmatrix}$
2. **Iterate:** for $t = 0, 1, \dots$ (until convergence):

$$\underline{p}^{(t+1)} = \underline{p}^{(t)} \mathbb{P}$$

3. **Stop** when: $|\underline{p}^{(t+1)} - \underline{p}^{(t)}| < \varepsilon$

Example

Ex: (Flow-model example..)

$$\begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 1 & 0 \end{bmatrix}$$

Initialize: $\underline{r}^{(0)} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$

Iteration :

$$\underline{r}^{(1)} = \underline{r}^{(0)}\mathbb{P} = [0.3333, 0.5000, 0.1667]$$

$$\underline{r}^{(2)} = \underline{r}^{(1)}\mathbb{P} = [0.4167, 0.3333, 0.2500]$$

$$\underline{r}^{(3)} = \underline{r}^{(2)}\mathbb{P} = [0.3750, 0.4583, 0.1667]$$

$$\underline{r}^{(4)} = \underline{r}^{(3)}\mathbb{P} = [0.4167, 0.3542, 0.2292]$$

$$\cdots\cdots$$

$$\underline{r}^{(37)} = \underline{r}^{(36)}\mathbb{P} = [0.4, 0.4, 0.2] = \underline{\pi}$$

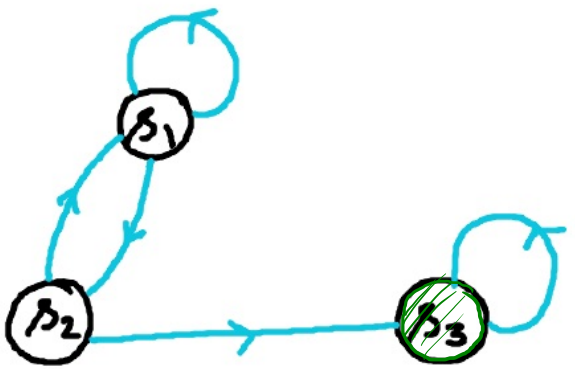
$$\underline{r}^{(38)} = \underline{r}^{(37)}\mathbb{P} = [0.4, 0.4, 0.2]$$

Stop: when $|\underline{r}^{(t+1)} - \underline{r}^{(t)}| < \varepsilon$, $\varepsilon = 0.00000001$

Convergence: Issues

1. **Spider Traps:**
   - all out-links are within the group
   - eventually absorbs all importance

Ex.



$$\mathbb{P} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 1 \end{bmatrix}$$

Initialize: $\underline{p}^{(0)} = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]$

Iteration:

$$\underline{p}^{(1)} = \underline{p}^{(0)}\mathbb{P} = [0.3333, 0.1667, 0.1667]$$

$$\cdots\cdots$$

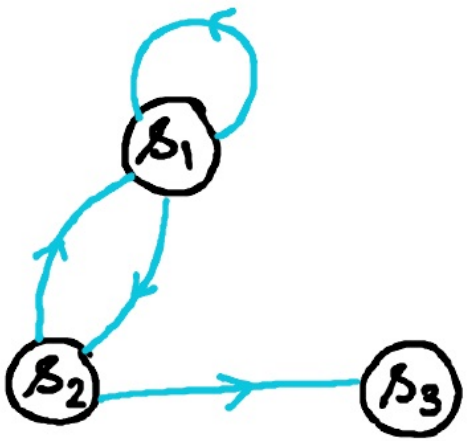
$$\underline{p}^{(45)} = \underline{p}^{(44)}\mathbb{P} = [0, 0, 1] = \underline{\pi}$$

Convergence: Issues

2. **Dead Ends/Dangling nodes:**

- Some pages are dead-ends which have no out-links
- they cause importance to leak-out

Ex.



$$\mathbb{P} = \begin{matrix} \\ p_1 \\ p_2 \\ p_3 \\ \\ \end{matrix} \begin{matrix} \begin{matrix} p_1 & p_2 & p_3 \end{matrix} \\ \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 \end{bmatrix} \\ \begin{matrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{matrix} \end{matrix}$$

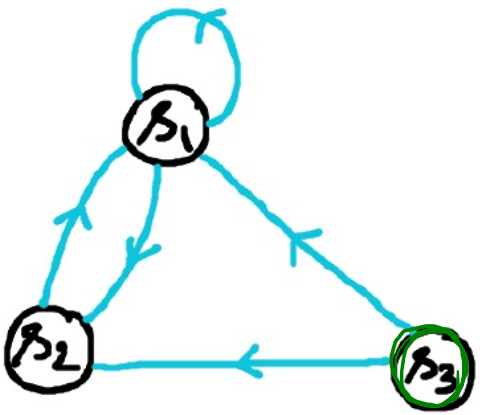
- $\mathbb{P}$ is not row-stochastic

Convergence: Issues

3. **Start Ends:**

   - Some starting pages have no in-links
   - they cause importance to leak-out

Ex.



$$\mathbb{P} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}$$

- **Stationary distribution:** $\underline{\pi} = [0.6667, 0.3333, 0]$

Solution to all Issues: Restart / Teleport

At each step, the random surfer has **two options**: **A** and **B**.

- *Probability of getting* ***option A*** *is* $1-\delta$:
  Teleport: Jump to one of the N pages at random with uniform probability
- *Probability of choosing* ***option B*** *is* δ:
  Usual Random Walk: Follow an out-link as in usual random walk

Common values of δ are in the range 0.8 to 0.9

Page Rank Equation-Corrected (Brin-Page '98):

$$p_j = (1-\delta)\frac{1}{N} + \delta \sum_{i \to j} \frac{p_i}{c_i}; \quad c_i : \text{out-degree of node } i$$

Google's Solution

In Matrix Form:

$$\underline{p} = (1-\delta)\left[\tfrac{1}{N}, \tfrac{1}{N}, \ldots, \tfrac{1}{N}\right] + \delta\, \underline{p}\mathbb{P}$$

Consequences of teleport:

- The state graph will be **strongly connected** and **aperiodic**
- Surfer will teleport out of a **spider-trap** within a few time steps. **Start-end** issue is also resolved since other pages can now reach to this page.
- **Dead-ends** have to resolved by "pre-processing" and then suitably modifying.

Google's Solution: Alternative Matrix Form

Google Matrix:

$$\mathbb{Q} = (1-\delta)\left[\tfrac{1}{N}\right]_{N\times N} + \delta\,\mathbb{P}$$

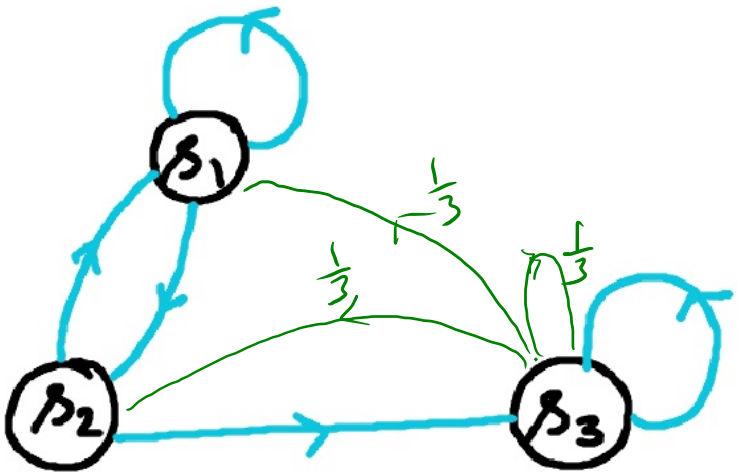
To Solve the system of equations: Apply Power Method

$$\underline{p} = \underline{p}\,\mathbb{Q}$$

In **Brin-Page '98**: δ is chosen as 0.85

Examples

Ex: (Spider trap)



$$\mathbb{Q} = \delta \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 1 \end{bmatrix} + (1-\delta) \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{7}{15} & \frac{7}{15} & \frac{1}{15} \\ \frac{7}{15} & \frac{1}{15} & \frac{7}{15} \\ \frac{1}{15} & \frac{1}{15} & \frac{13}{15} \end{bmatrix} \quad (\text{using } \delta = 0.8)$$

Initialize: $\underline{p}^{(0)} = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$

Iteration :

$$\underline{\pi} = \left[\frac{7}{33}, \frac{5}{33}, \frac{21}{33}\right].$$

Personalized Page Rank

- **Page rank equation:**

$$\underline{p} = (1-\delta)\underline{t} + \delta\, \underline{p}\mathbb{P} \Longrightarrow \underline{p} = (1-\delta)\underline{t}(I - \delta\mathbb{P})^{-1}$$

where $\underline{t} = \left[\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}\right]$: uniform distribution, I: $N \times N$ identity matrix

- For **personalized page rank:** use non-uniform teleportation distribution,

  i.e. at any time step 'teleport to a set of web-pages' according to a pmf $\underline{s}$,

  personalized page rank equation:

$$\underline{p} = (1-\delta)\underline{s} + \delta\, \underline{p}\mathbb{P}$$

References