PageRank: An Eigenvector Problem

Minh Tran • Komlan Wussinu • Tony Nielsen

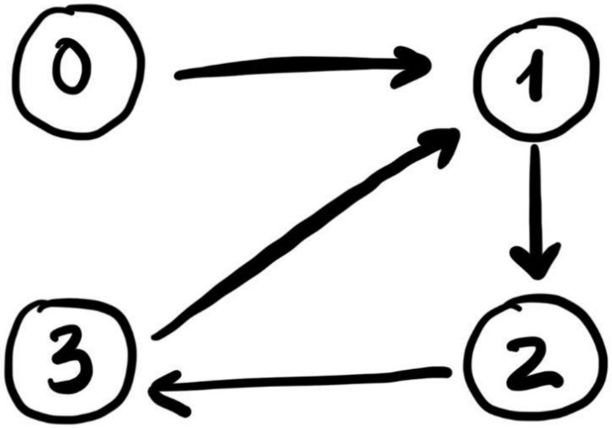
What is the PageRank algorithm?

- **Larry Page & Sergey Brin** (1998)
- Determine a page's **importance** based on link structure
- A page is important if other **important pages linked** to it
- The secret sauce of the world's **most powerful** search engine
- I.e. Forbes, academic pages, etc.

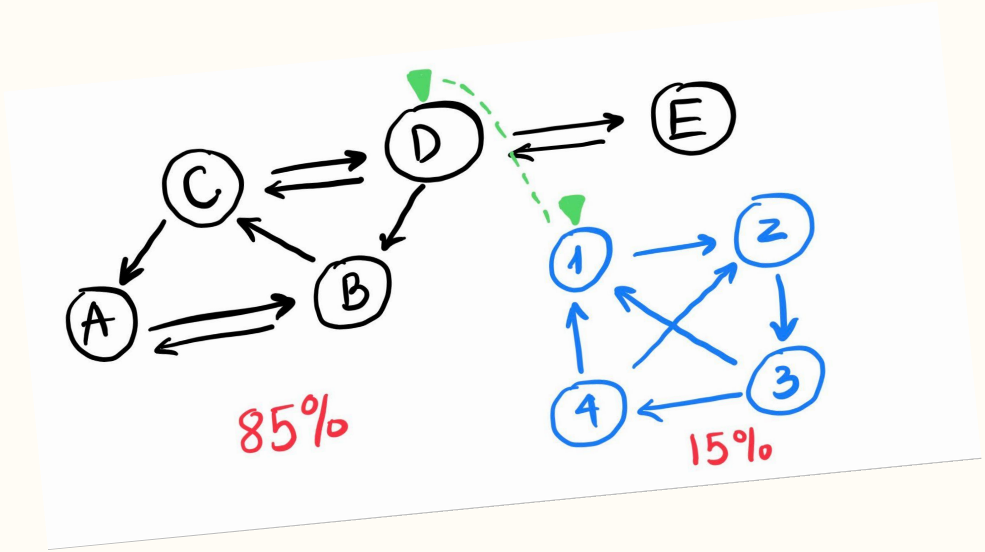
The Web as a Directed Graph

Nodes = **pages** | Edges = **links** | Adjacency Matrix: column = **outgoing**, row = **incoming**

Out-degree: **# of outgoing link**



	0	1	2	3
0	0	0	0	0
1	1	0	0	1
2	0	1	0	0
3	0	0	1	0

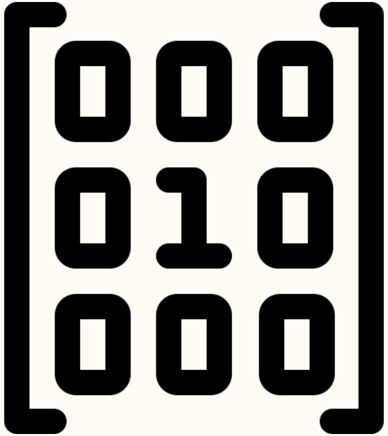


Random Surfer Model

Damping factor: typically 0.85

- 85% follow links
- 15% hop to random pages
- More on this later ...

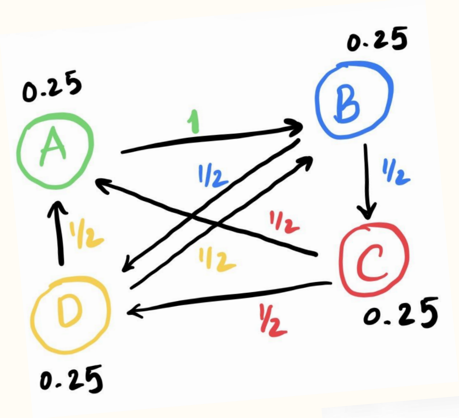
The PageRank Equation

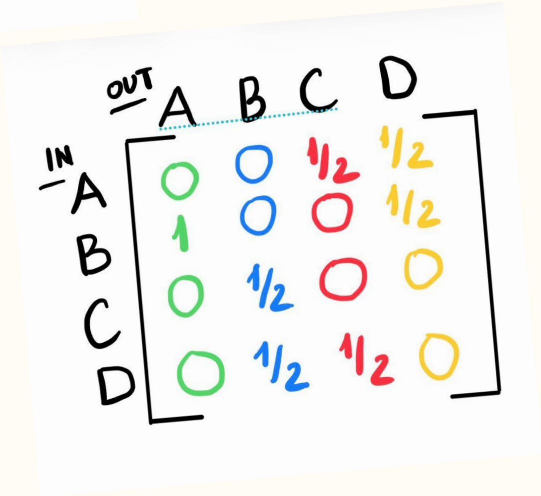


Transition Matrix

Google Matrix

PageRank Vector

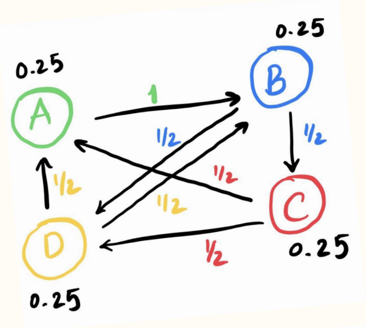




Transition Matrix M

$$M_{ij} = 1 / d_j$$

- Page j links to page i
- Column-stochastic (sums to 1)
- $d_j > 0$: out-degree of page j_j



$$
\begin{array}{c|cccc}
\text{IN} \backslash \text{OUT} & A & B & C & D \\
\hline
A & 0 & 0 & 1/2 & 1/2 \\
B & 1 & 0 & 0 & 1/2 \\
C & 0 & 1/2 & 0 & 0 \\
D & 0 & 1/2 & 1/2 & 0
\end{array}
$$

For each entry : $G_{ij} = \alpha M + \frac{1-\alpha}{n}$

$$G_{11} = 0.85(0) + \frac{1-0.85}{4}$$

$$G = \begin{bmatrix} 0.0375 & 0.0375 & 0.4625 & 0.4625 \\ 0.8875 & 0.0375 & 0.0375 & 0.4625 \\ 0.0375 & 0.4625 & 0.0375 & 0.0375 \\ 0.0375 & 0.4625 & 0.4625 & 0\ 0375 \end{bmatrix}$$

Google Matrix G

$G = \alpha M + (1-\alpha)/n\ (ee^T)$

- **α - damping factor** : 0.85
- 2nd term:
  - **Damping adjustment** or teleportation
  - **Uniform** probability

$(1-\alpha)/n\ (ee^T)$

- **Handle dead ends**
  - Stuck = sum < 1
- **Closed-circle traps**
  - Spam farming
- **Connection & Speed**
  - Not fully connected
  - Huge computing time
- **Humanizing the algorithm**
  - 85% following links
  - 15% reset

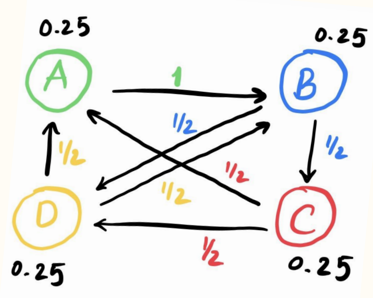
PageRank as an Eigenvector Problem

$R_{k+1}\,\lambda = G \times R_k$

- **Power iteration**
  - ...(G(G(GxR)))
  - $\lim_{n\to\infty} G^nR = R$
- **Eigenvalue λ = 1**
  - **R:** rank scores = probabilities sums to 1
  - ➔ λ forced to be 1
- **Self-consistent**
  - Stable ranking system

$$G = \begin{bmatrix} 0.0375 & 0.0375 & 0.4625 & 0.4625 \\ 0.8875 & 0.0375 & 0.0375 & 0.4625 \\ 0.0375 & 0.4625 & 0.0375 & 0.0375 \\ 0.0375 & 0.4625 & 0.4625 & 0.0375 \end{bmatrix} \times \overset{R_0}{\begin{bmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix}}$$

$$R_1 \approx \begin{bmatrix} 0.25 \\ +0.36 \\ +0.14 \\ +0.25 \end{bmatrix} = 1$$



Python Demonstration

https://github.com/minhtran021999/PageRank-Project.git

PAGE RANK YOU FOR YOUR ATTENTION!

The End.