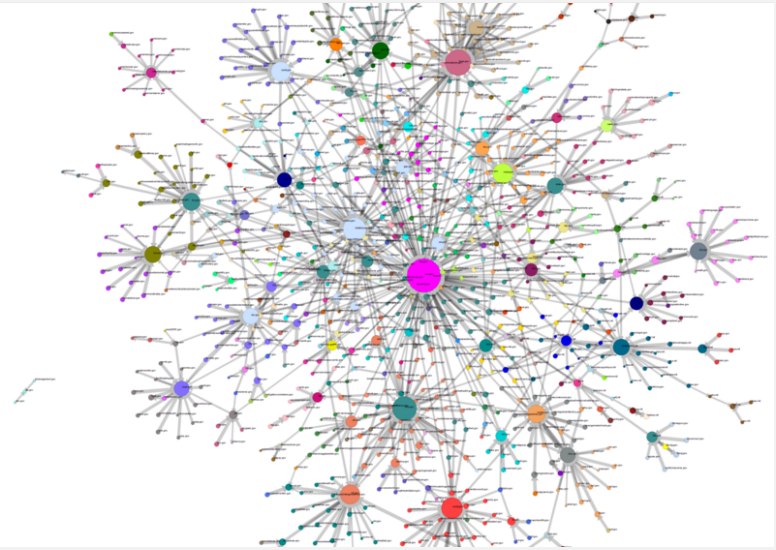

Google
PageRank

The Web as a Graph

- The web can be seen as a **directed graph**:
  - Nodes = web pages
  - Edges = hyperlinks



- The web pages are not equally important
  - **Goal**: rank the pages using the web graph link structure, that is, compute the importance of the nodes in the web graph

Links as votes

- Idea: **Links as votes**
  - We can think of in-coming links as votes
  - A page is more important if it has more in-coming links

- But not all the in-links are the same
  - Links from important pages should count more
  - **Recursive idea**: the importance of a page depends on the importance of the pages that link to it
  - The importance of a page gets equally split among its out-links

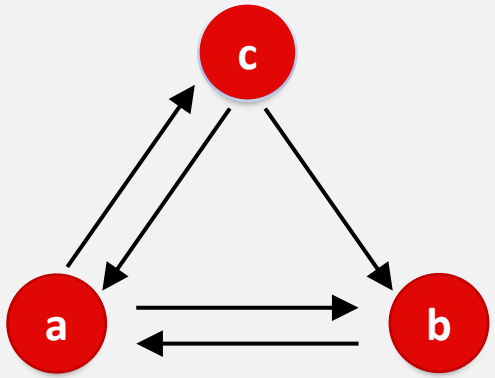
PageRank: the Flow Model

- The importance of a page is given by the number of pages that point to it and by the importance of those pages.

- Consider d_j to be the number of out-links of node j.

  Then, we can define the **rank** r_i for node i

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



Flow equations:

$$r_a = \frac{r_c}{2} + rb$$

$$r_b = \frac{r_a}{2} + \frac{r_c}{2}$$

$$r_c = \frac{r_a}{2}$$

We can use Gaussian elimination to solve this system of linear equations, but that is a bad idea.

PageRank: Matrix Formulation

- Stochastic **adjacency matrix** M
  - $M_{ij} = \begin{cases} \frac{1}{d_j} & if\ j \to i \\ 0 & otherwise \end{cases}$

    where d_j are the out-links of page j

    M is a column stochastic matrix: columns sum up to 1
- **Rank vector** r
  - r_i is the rank of page i and $\sum_i r_i = 1$
- The flow equations can be written as

$$r = Mr$$

Connection to Random Walk

- Imagine a **random web surfer**
  - At any time $\boldsymbol{t}$, the random surfer is on some page $\boldsymbol{j}$
  - At time $\boldsymbol{t+1}$, the random surfer follows an out-link from $\boldsymbol{j}$ uniformly at random, ending up on some page $\boldsymbol{i}$ linked from $\boldsymbol{j}$
  - Process repeats indefinitely

- Where is the surfer at time $t+1$?
  - Let $\boldsymbol{p(t)}$ be the vector whose j^{th} coordinate is the probability that the surfer is at page j at time t ($p(t)$ is a probability distribution over the web pages)
  - So, $\boldsymbol{p_i(t+1) = \sum_{j \to i} \frac{p_j(t)}{d_j}}$ or $\boldsymbol{p(t+1) = Mp(t)}$

Connection to Random Walk

- Suppose the random walk reaches a state
  - $\boldsymbol{p(t+1) = Mp(t) = p(t)}$

  Then $\boldsymbol{p(t)}$ is the **stationary distribution** of the random walk

- But our original rank vector $\boldsymbol{r}$ satisfies $\boldsymbol{r = Mr}$, so the PageRank vector $\boldsymbol{r}$ is a **stationary distribution** of the random walk

- Moreover, recalling the definitions of eigenvector and eigenvalue, from $1 \cdot r = M \cdot r$, we have that the **rank vector $\boldsymbol{r}$** is an **eigenvector** of the stochastic adjacency matrix $\boldsymbol{M}$ (with eigenvalue 1)

Summarizing

- **PageRank**:
  - Measures importance of nodes in a graph using the link structure of the web
  - It solves the flow equation $r = Mr$
  - r can be viewed both as the stationary distribution of a random walk over the graph and as the eigenvector of M corresponding to eigenvalue 1

- How to solve? **Power iteration**

Power Iteration Method

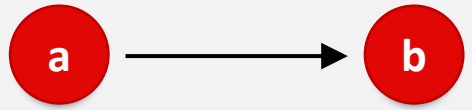
- Given a web graph with n nodes, the **power iteration** method is an iterative procedure

- Initialize: $r^{(0)} = \left[\frac{1}{n}, \ldots, \frac{1}{n}\right]^T$
- Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
- Until: $\left\| r^{(t+1)} - r^{(t)} \right\|_1 < \varepsilon$

where $\|x\|_1 = \sum_{i=1}^{n} |x_i|$ is the L_1 norm

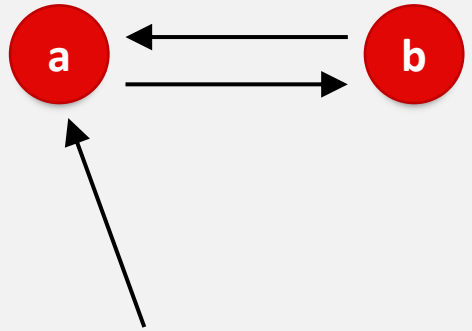
Problems

- **Dangling nodes (or dead ends)**: some pages have no out-links
  - **Solution**: modify the matrix M by adding artificial links that uniformly connect dangling pages to all the pages; with probability 1 the random surfer teleport to a random page



- **Spider traps**: all out-links are within the group
  - **Solution**: at each time step, the random surfer has two options:
    1) With probability β, follow a link at random
    2) With probability $1 - \beta$, jump to a random page

    The random surfer will teleport out of spider traps in a finite number of steps



The Google Solution

- **PageRank equation** (Brin-Page, 1998): $r_i = \sum_{j \to i} \beta \frac{r_j}{d_j} + (1 - \beta) \frac{1}{n}$
- **The Google matrix G**: $G = \beta M + (1 - \beta) \left[\frac{1}{n}\right]_{n \times n}$
- **Power iteration**:
  - Initialize: $r^{(0)} = \left[\frac{1}{n}, \ldots, \frac{1}{n}\right]^T$
  - Iterate: $r^{(t+1)} = G \cdot r^{(t)}$
  - Until: $\left\| r^{(t+1)} - r^{(t)} \right\|_1 < \varepsilon$

β is usually taken between 0.8 and 0.9

Why does it work?

- G needs to be positive, stochastic, irreducible and aperiodic.
- The **Perron-Frobenius theorem** for positive squared matrices, combined with stochasticity guarantees that there exists the dominant eigenvalue of G (strictly greater than any other eigenvalue of G in absolute value) and that it is 1. Given that our matrix is irreducible, the P-F theorem guarantees that the dominant eigenvalue is simple and that the corresponding eigenvector has positive components. This proves the existence of a unique positive solution.
- It can be proved that the power iteration method converges to the principal eigenvector of G

Extensions: Personalized PageRank

- PageRank assumes that all the internet users follow the same probabilistic rule when deciding which webpage to go to. In reality, it makes sense to include personalized information into our algorithm, that is, we want to rank the webpages by taking each individual's preference into consideration.
- Consider the **personalization vector $\boldsymbol{p}$** for each user, that is the (pre-specified) probabilistic distribution of visiting each of the n webpages for that particular user (typically contains many zeros).
- The **teleportation matrix $\boldsymbol{P}$** for each user is an n times n matrix with each column corresponding to the user's personalization vector p.

Personalized PageRank

- We can still use **power iteration**
- The update formula will be:

$$r^{(t+1)} = \left(aM + bP + c\left[\frac{1}{n}\right]_{n \times n}\right) \cdot r^{(t)}$$

where the matrix of all $\frac{1}{n}$'s is added to ensure aperiodicity and irreducibility

and $a + b + c = 1$

- Or equivalently, since r sums up to 1

$$r^{(t+1)} = aM \cdot r^{(t)} + b \cdot \mathrm{p} + \mathrm{c}\left[\frac{1}{n}\right]_{n \times 1}$$

where p is the n times 1 personalization vector

Extensions: Random Walk with Restart

- Same idea as Personalized PageRank, but the personalization vector p will contain all zeros except for the component corresponding to the starting page, which is going to be 1
- Can still use power iteration and the update formula:

$$r^{(t+1)} = aM \cdot r^{(t)} + b \cdot \mathrm{p} + \mathrm{c}\left[\frac{1}{n}\right]_{n \times 1}$$

- It gives the closeness of different nodes in the graph (typical application: recommendation systems)

Applications

- PageRank is now regularly used far beyond web search
- Some examples include:
  - Recommendation systems (ex. Social media)
  - Scientific research and academia, to quantify the scientific impact of publications or authors
  - Biology, for the analysis of protein networks
  - In neuroscience, the PageRank of a neuron in a neural network has been found to correlate with its relative firing rate
- … or to rank Wikipedia pages (practical part)

Some references

- P. Berkhin (2005). “A survey on PageRank Computing”
- S. Brin, L. Page (1998). “The anatomy of a large-scale hypertextual Web search engine”
- L. Page, S. Brin, R. Motwani, T. Winograd (1999). “The PageRank citation ranking: Bringing order to the Web”
- C. Consonni , D. Laniado, A. Montresor (2019). “WikiLinkGraphs: A complete, longitudinal and multi-language dataset of the Wikipedia link networks”
- P. Chen, H. Xie, S. Maslov, S. Redner (2007). “Finding scientific gems with Google”
- G. Ivan, V. Grolmusz (2011). “When the Web meets the cell: using personalized PageRank for analyzing protein interaction networks”
- D. Banky, G. Ivan, V. Grolmusz (2013). “Equal opportunity for low-degree network nodes: a PageRank-based method for protein target identification in metabolic graphs”
- J. M. Fletcher, T. Wennekers (2017). "From Structure to Activity: Using Centrality Measures to Predict Neuronal Activity"