



Social Data Management

Node and Link Analysis

Silviu Maniu¹

January 7th, 2019

¹Université Paris-Sud

Estimating Node Worth

Nodes on the Web: pages (sites, Wikipedia, . . .), users (Twitter, Facebook), etc.

To use/find the nodes that are more “interesting” than others, we have to **estimate the worth of each node**.

Can be used combined with textual (or profile) information to retrieve content in **information retrieval** – but also the *links* between information are important

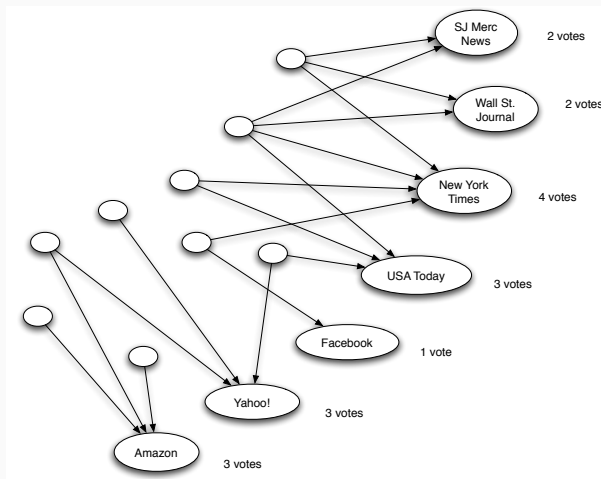
Estimating Node Worth

Node centrality can provide a way to *rank nodes*

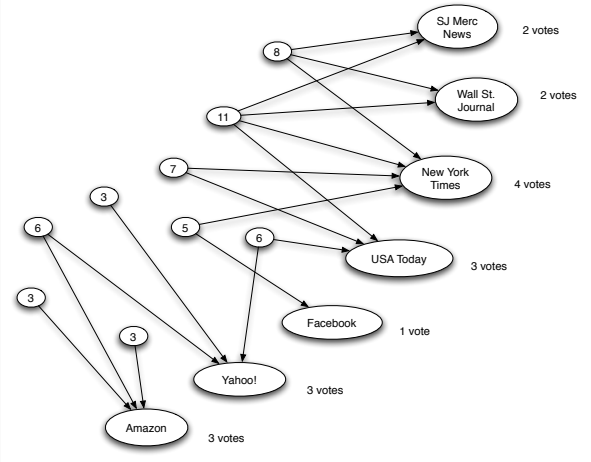
Some possibilities:

- degree centrality
- weighted degree centrality

Degree Centrality



Weighted Degree Centrality



Repeated Improvement

Principle of Repeated Improvement: the pages which link to high-worth pages are high-worth themselves

Suggests an two-way relationship:

- nodes which are relevant are called **authorities**, and are linked to by good nodes
- nodes which link to many other high-authority nodes are called **hubs**

Hubs and Authorities

Each node $i \in V$ has attached an **authority score** a_i and a **hub score** h_i

Each score is update **in relation to the other**.

HITS Algorithm

1. **Initialize:** for each $i \in V$ $a_i = 1$, $h_i = 1$
2. **Authority Update Rule:** the authority score of a page is the sum of hub scores of the *incoming* neighbours

$$a_i = \sum_{j \in I(i)} h_j$$

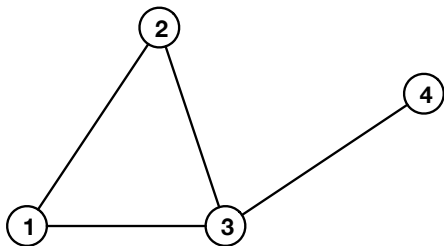
3. **Hub Update Rule:** the hub score of a page is the sum of the authority scores of the *outgoing* neighbours

$$h_i = \sum_{j \in O(i)} a_j$$

4. **Normalization** the sum of all hub (or authority) scores must be the same between steps

In practice, the algorithm is repeated for a fixed number of steps k

Applying HITS – Example



HITS: Matrix View

In **matrix form**, at time k , the vectors \mathbf{a} and \mathbf{h} have the following forms:

$$\mathbf{a}^{(k)} = A^T \mathbf{h}^{(k-1)} \quad \mathbf{h}^{(k)} = A \mathbf{a}^{(k-1)}$$

We can see that, e.g.:

$$\mathbf{h}^{(k)} = AA^T \mathbf{h}^{(k-2)},$$

which leads to the general formulas:

$$\mathbf{h}^{(k)} = (AA^T)^k \mathbf{h}^{(0)}$$

$$\mathbf{a}^{(k)} = (A^T A)^{k-1} A^T \mathbf{h}^{(0)}$$

Applying spectral analysis, it can be shown that the normalized values **converge** when $k \rightarrow \infty$

- computes **two scores** for each nodes, which might be hard to interpret
- it is usually used for particular **queries**, and only a subset of pages
- it is not usually used in current search engines

Table of contents

Hubs and Authorities

PageRank

Link Prediction

PageRank: Ranking Nodes in A Graph

Definition 1: The important nodes are the nodes that are linked to by other important nodes (**recursive**).

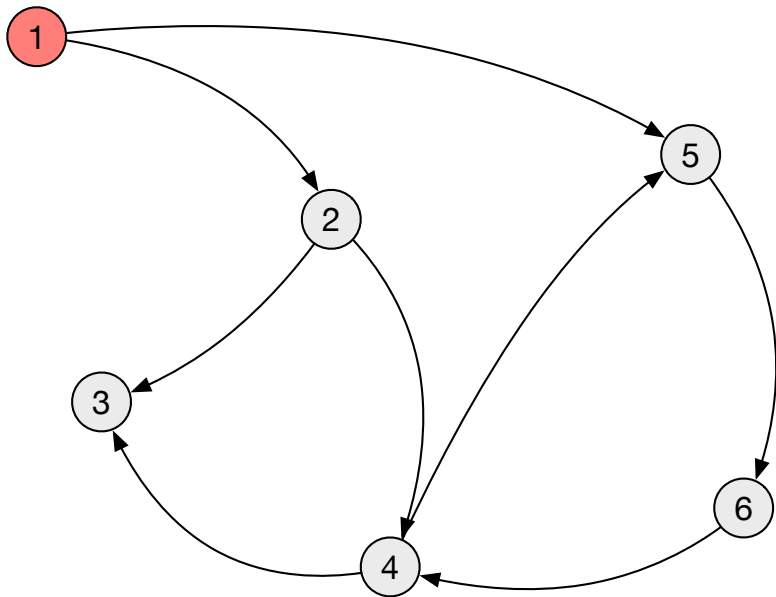
Definition 2 – the **random surfer model**, where the surfer **walks** on the graph:

1. the surfer starts at a node (e.g., Google) and chooses randomly an outgoing node (e.g., a page in the search results),
2. the surfer behaves in the same manner for other nodes,
3. at each step the surfer has a probability $1 - \alpha$ (damping factor) of jumping elsewhere randomly.

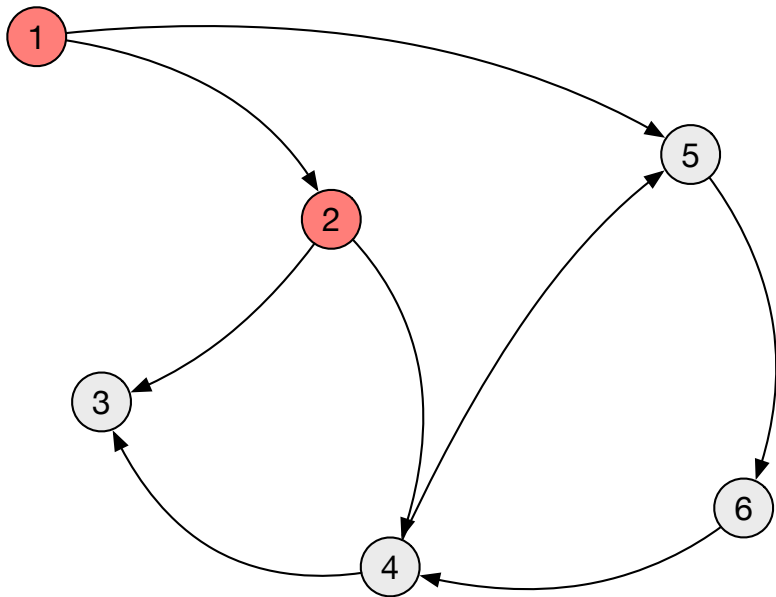
The importance of a page = the **stationary probability** that the surfer is on a page at time ∞ .

The two definitions are **equivalent**.

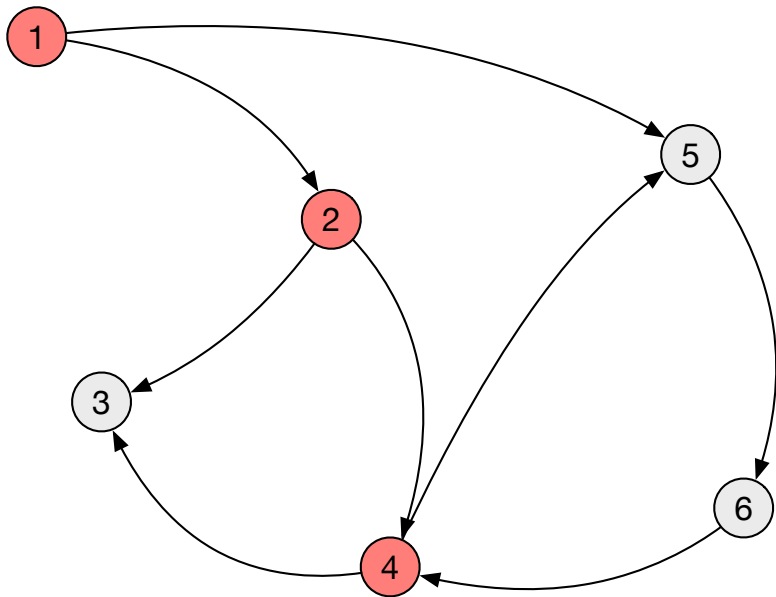
PageRank: Random Surfer



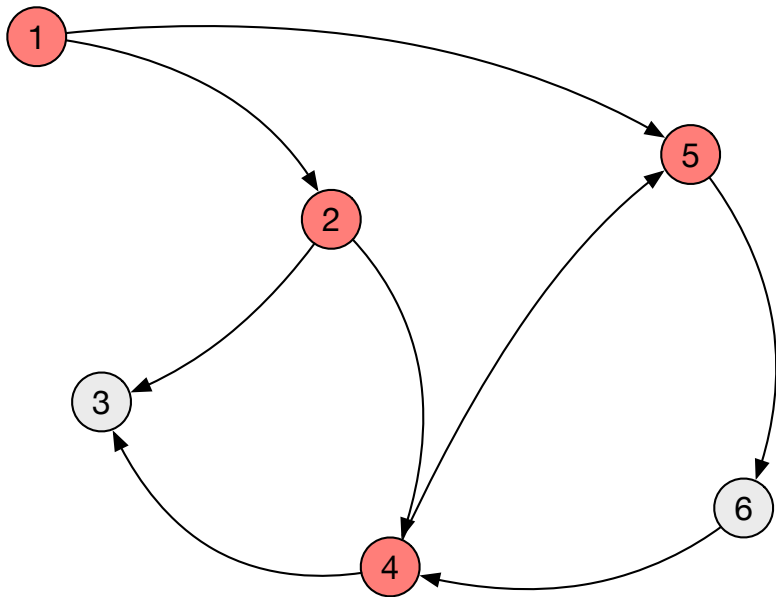
PageRank: Random Surfer



PageRank: Random Surfer



PageRank: Random Surfer



PageRank Equation and Algorithm

For a graph G having n nodes, where each node i has the incoming neighbours I_i and outgoing neighbours O_i :

$$p(i) = \alpha \sum_{j \in I_i} \frac{p(j)}{|O_j|} + \frac{1 - \alpha}{n}.$$

Algorithm for computing $p(i)$:

1. start with initial values of $p(i) = \frac{1}{n}$,
2. iteratively apply the equation for each node i ,
3. stop when the probabilities converge (**stationary**).

Monte-Carlo approximation: simulate N walks and take $p(i) = \frac{v_i}{N}$, where v_i number of visits of page i among N walks.

We need to define a transition matrix M and a teleportation vector \mathbf{t} .

The transition matrix is a **stochastic matrix** where:

$$m_{ij} = \frac{a_{ij}}{|O_i|},$$

where a_{ij} is the corresponding entry in the adjacency matrix; it is stochastic because $\sum_j m_{ij} = 1$.

The teleportation vector \mathbf{t} is also stochastic:

$$\mathbf{t} = \left(1/n \quad \dots \quad 1/n\right)^\top$$

The PageRank vector \mathbf{p} at step k is then defined as:

$$\mathbf{p}^{(k)} = \alpha M \mathbf{p}^{(k-1)} + (1 - \alpha) \mathbf{t}$$

This can be proved it **converges** to a known value $\mathbf{p}^{(\infty)}$, the **stationary distribution**:

$$\mathbf{p}^{(\infty)} = (I - \alpha M)^{-1} (1 - \alpha) \mathbf{t}$$

Variants of PageRank

Depending where the surfer teleports with probability $1 - \alpha$, we have different variants of PageRank:

- **classic PageRank**: the surfer can jump to any node.
- **personalized PageRank**: the surfer can only jump to their start page

$$\mathbf{t}_j = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \end{pmatrix}^\top$$

- **topic-sensitive PageRank**: the surfer can only jump to a set of same-topic pages

$$\mathbf{t} = \begin{pmatrix} 0 & 1/l & 1/l & \dots & 0 \end{pmatrix}^\top$$

Table of contents

Hubs and Authorities

PageRank

Link Prediction

The Link Prediction Problem

Social networks are **evolving**, and new relationships (links) appear all the time

Link Prediction Problem: predict which links are more likely to appear in a social network

Assumes that links can be predicted via analysis based only on the social network itself

Applications:

- new link **recommendation** (e.g., new friends)
- missing **link inference**
- analyzing **network evolution**

Link Scoring Function

We want to “guess” the **score** of potential links for a graph $G = (V, E)$, i.e., a function defined on the missing links $E' = (V \times V) \setminus E$:

$$\text{score} : E' \rightarrow \mathbb{R}^+$$

For a given i , $\text{score}(i, j)$ established a **ranking** of all (unliked) nodes j relative to i – best scores are the most likely new links

How can we define the score function using only the properties intrinsic to the network?

Node Neighbourhood Scores

- **Common Neighbours**, most straightforward just counts the number of common neighbours:

$$\text{score}(i, j) = |N(i) \cap N(j)|$$

- **Jaccard coefficient**, computes the “similarity” between the neighborhood sets

$$\text{score}(i, j) = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|}$$

- **Preferential attachment**, the score is proportional to the degrees of each node:

$$\text{score}(i, j) = k_i k_j$$

Path-Based Scores

- **Inverse Distance**, the score is inversely proportional to the distance between two nodes

$$\text{score}(i, j) = 1/d_{ij}$$

- **Katz**, where the score is a weighted sum of all the path between i and j

$$\text{score}(i, j) = \sum_{l=1}^{\infty} \beta^l |\text{paths}_{i,j}^{(l)}|,$$

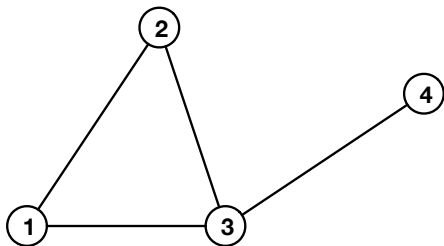
where $\beta \in (0, 1)$

Random Walk-Based Scores

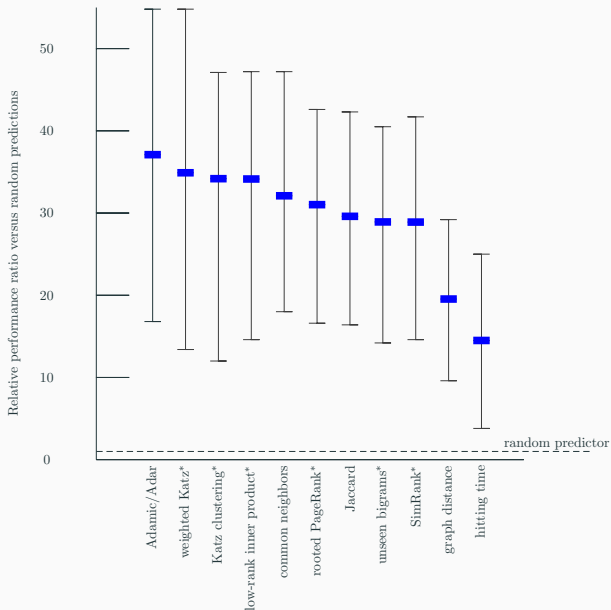
- **Hitting time**, $\text{score}(i, j) = H_{i,j}$ the time it takes a random walk from i to reach j
- **Personalized PageRank**, generally any PageRank-related measure in which the teleportation vector is rooted at i
- **SimRank**, a recursive definition based on the score of neighbours

$$\text{score}(i, j) = \gamma \frac{\sum_{a \in N(i)} \sum_{b \in N(j)} \text{score}(a, b)}{k_i k_j}$$

Applying the Scores – Example



Performance of Link Prediction



Acknowledgments

Figures in slides 4 and 5 are taken from the book “Networks, Crowds, and Markets”, D. Easley and J. Kleinberg, Cambridge University Press, 2010.

The figure in slide 29 is taken from [Liben-Nowell and Kleinberg, 2003].



Kleinberg, J. M. (1999).

Authoritative sources in a hyperlinked environment.

J. ACM, 46(5).



Liben-Nowell, D. and Kleinberg, J. (2003).

The link prediction problem for social networks.

In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*.



Page, L., Brin, S., Motwani, R., and Winograd, T. (1998).

The PageRank citation ranking: Bringing order to the Web.

In *Proceedings of the 7th International World Wide Web Conference*.