Google PageRank algorithm

(Link analysis)

Parsa Tasbihgou
Prof. A. Jamshidi
Department of computer science and mathematics,
University of Tehran
July, 2021

Outline

- First search methods and there difficulties
- PageRank problem
- PageRank algorithm
- Probability matrices
- HITS algorithm
- Other applications: spread modeling

First search algorithms and difficulties

- Before PageRank, search algorithms used plain keyword matching, or a combination with citation counters.
- PageRank prioritizes results from keyword matching, and integrates citation measures in the structure of the webgraph.
- The ranking produced by PageRank is preprocessed and there is no computation overhead at query time.

PageRank algorithm

The random surfer model

Let G be a directed graph (equivalently a network) and S some initial node in G chosen at random.

let Bob be a “random surfer” who starts from S and randomly choses an outgoing edge and moves to the other endpoint of that edge; he keeps doing that without ever moving backwards. Eventually Bob gets tired and starts over from some random node, the probability of Bob not getting bored at each node and continue surfing is that node’s damping factor denoted with d. The ranking of the nodes of G is the probability of Bob visiting that node (higher probability means better rank).

PageRank problem

Pagerank problem

Let A be a node in some graph G with n vertices.

and let T_1, T_2, ..., T_k be vertices having outgoing edges to A, so A has incoming edges from T_1, T_2, ..., T_k, also let C(v) denote the number of outgoing edges from v, and d be the damping factor.

the PageRank of A is defined as follow:

$$PR(A) = \frac{(1-d)}{n} + d(\frac{PR(T_1)}{C(T_1)} + \ldots + \frac{PR(T_k)}{C(T_k)})$$

PageRank solution

Let H = ($h_{i,j}$) where $h_{i,j} = \begin{cases} \frac{1}{L(v_j)}, & v_j \text{ is connected to } v_i \\ 0, \text{ otherwise} \end{cases}$ it is easy to see that this definition of H is

equivalent to $H = (D^{-1}A)^T$ where D is the diagonal out-degree matrix and A is the basic adjacency matrix.

Let PR = ($PR(v_1)$, $PR(v_2)$, ..., $PR(v_n)$).

And finally let e be an n-vector with all of its entries equal to 1.

Now we define the google matrix as $G = dH + ((1-d)e \times V^{(0)})$ where $V^{(0)}$ is personalization vector that defines the probability distribution of selection of the starting vertex, $V^{(0)}$ is initially set to $(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$. It is easy to verify that $PR^{(k+1)} = PR^{(k)} G$, we continue these iteration until PR converges.

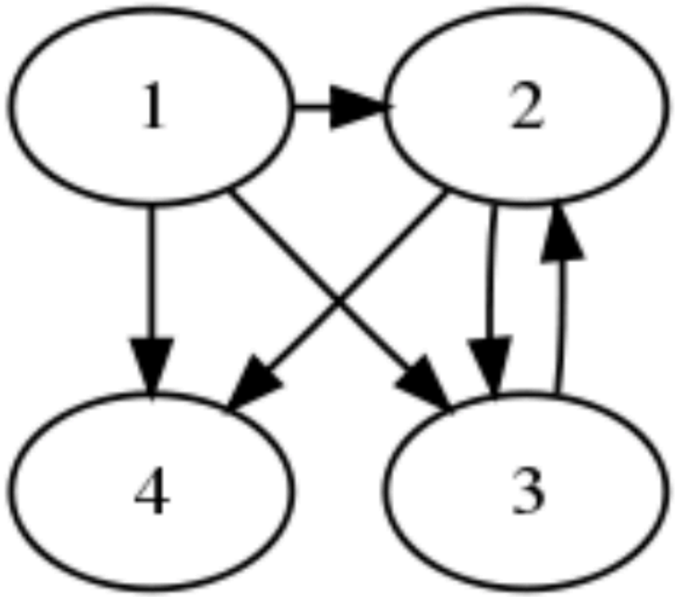
EXAMPLE:

A

0	1	1	1
0	0	1	1
0	1	0	0
1	1	1	1

H

0	0.33	0.33	0.33
0	0	0.5	0.5
0	1	0	0
0.25	0.25	0.25	0.25



D

3	0	0	0
0	2	0	0
0	0	1	0
0	0	0	4

D_inv

0.33	0	0	0
0	0.5	0	0
0	0	1	0
0	0	0	0.25

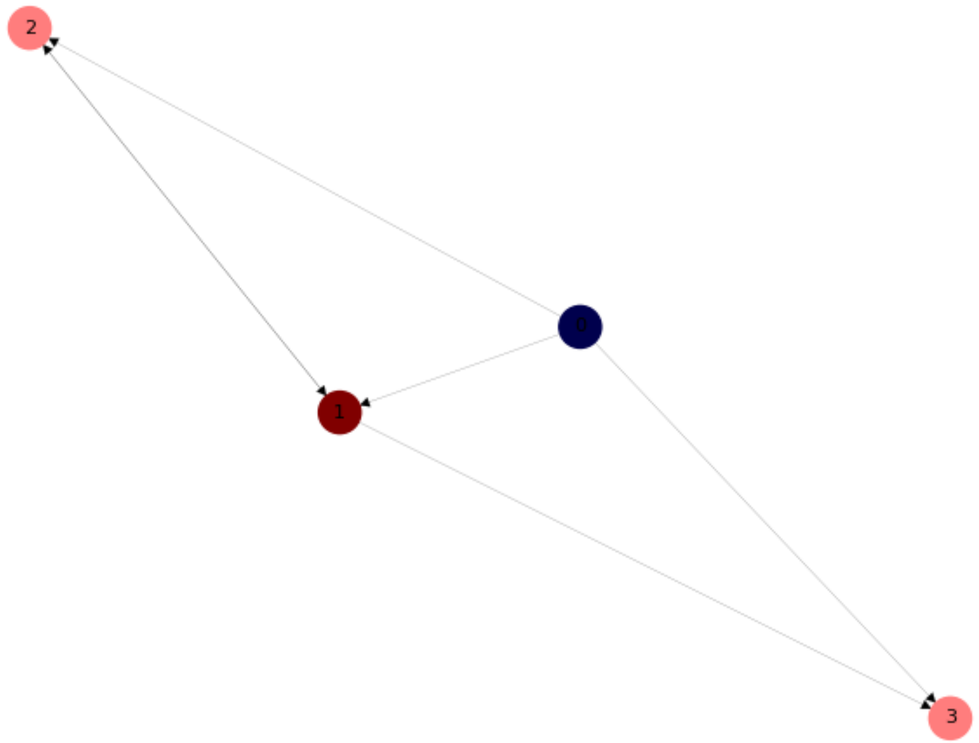
G

0.0375	0.318	0.318	0.318
0.0375	0.0375	0.4625	0.4625
0.0375	0.8875	0.0375	0.0375
0.25	0.25	0.25	0.25

vG = v, therefore v is the eigenvector corresponding to eigenvalue 1.

Using any method (i.e. inverse power method) the eigenvector is calculated:

$v = (0.0967, 0.3606, 0.2712, 0.2712)^T$ so the PageRank method generates the following ranking:



Index	Rank
0	**1**
1	4
2	3
3	3

Probability matrices

A (left) stochastic matrix is a real non-negative square matrix that the sum of each of its rows is equal to 1.

Let M be a probability matrix.

1. It is easy to see that one of its right eigen pairs is (1, e), since the sum of each row is 1.

2. From Gershgorin disks theorem we can prove that the spectral radius of M is 1, therefore 1 and the corresponding eigenvectors are dominant eigen pairs.

A stationary probability vector π for M, is a probability distribution such that it doesn't change by applying the probability matrix that is:

$$\pi \times M = \pi$$

π is the left eigenvector corresponding to 1.

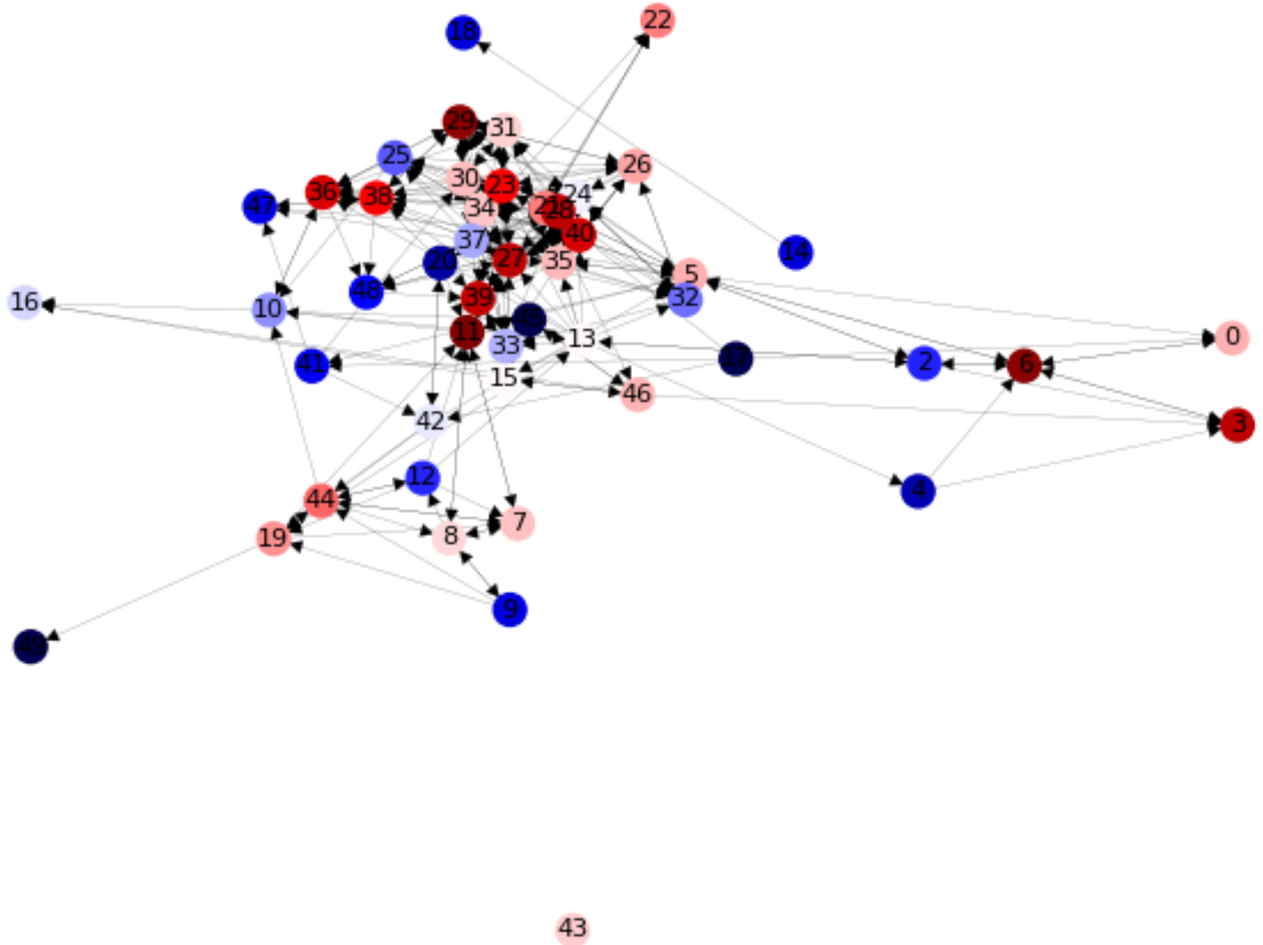
Uniqueness of stationary vectors

Stationary probability vectors of Probability matrices don’t generally need to be unique, but they are unique when the probability matrix is irreducible (or more specifically: with strictly positive entries). A stochastic matrix is **irreducible** if and only if the corresponding weighted directed graph is strongly connected.

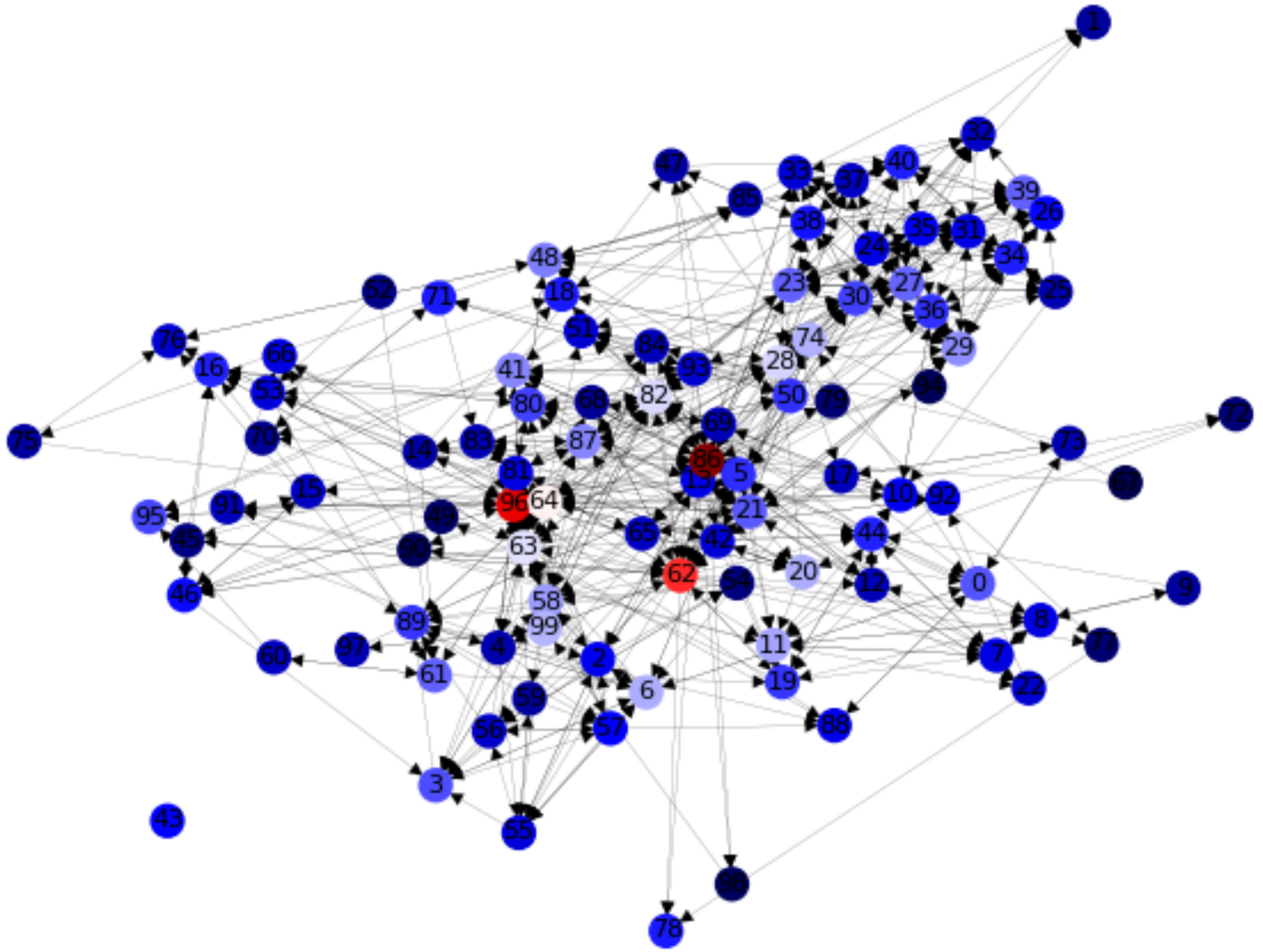
Now that we have stablished the relation between PageRank the eigenvectors, we can easily calculate the PageRank vector using eigen algorithms for example the inverse power method.

Results

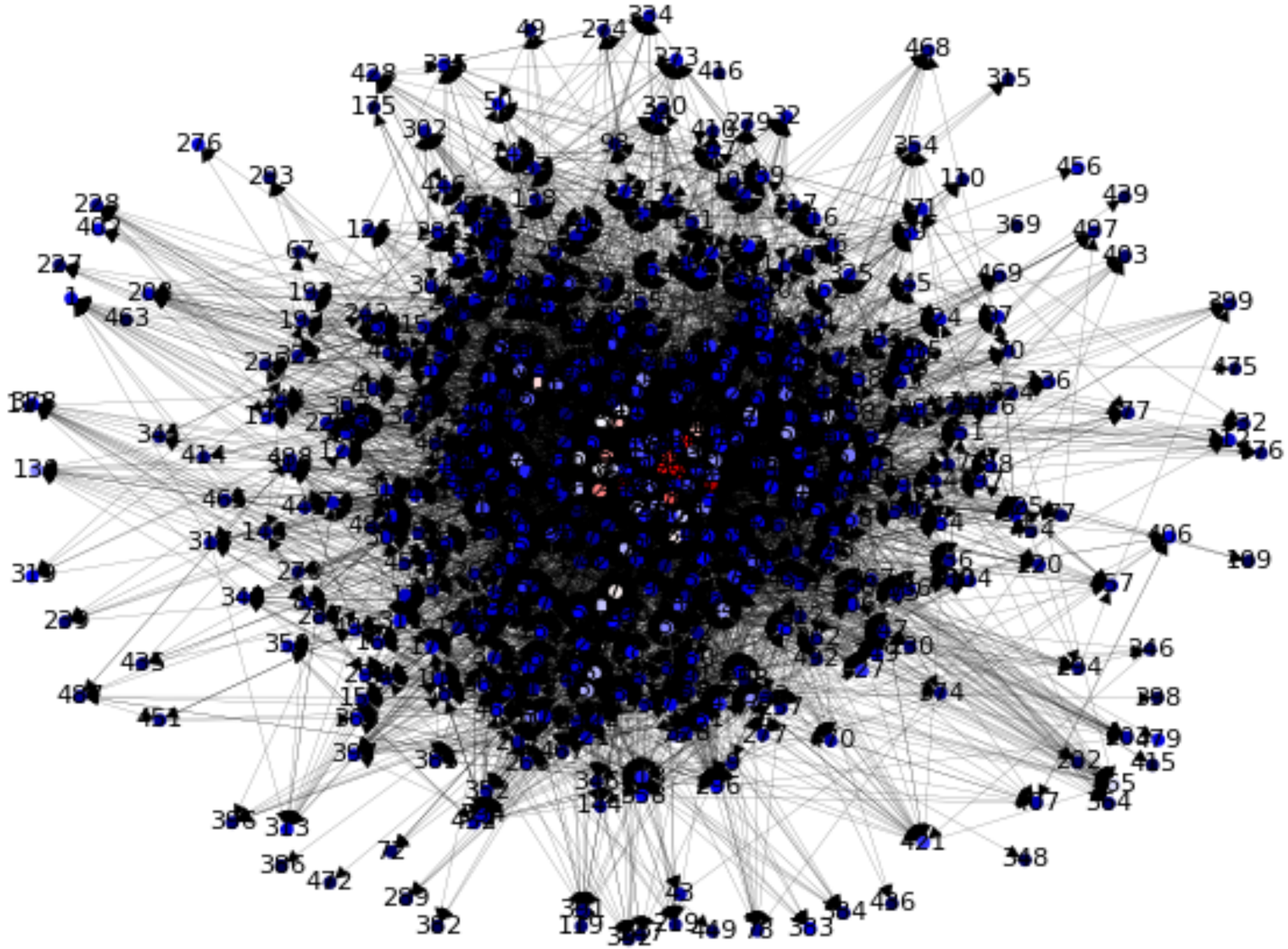
50 node data set



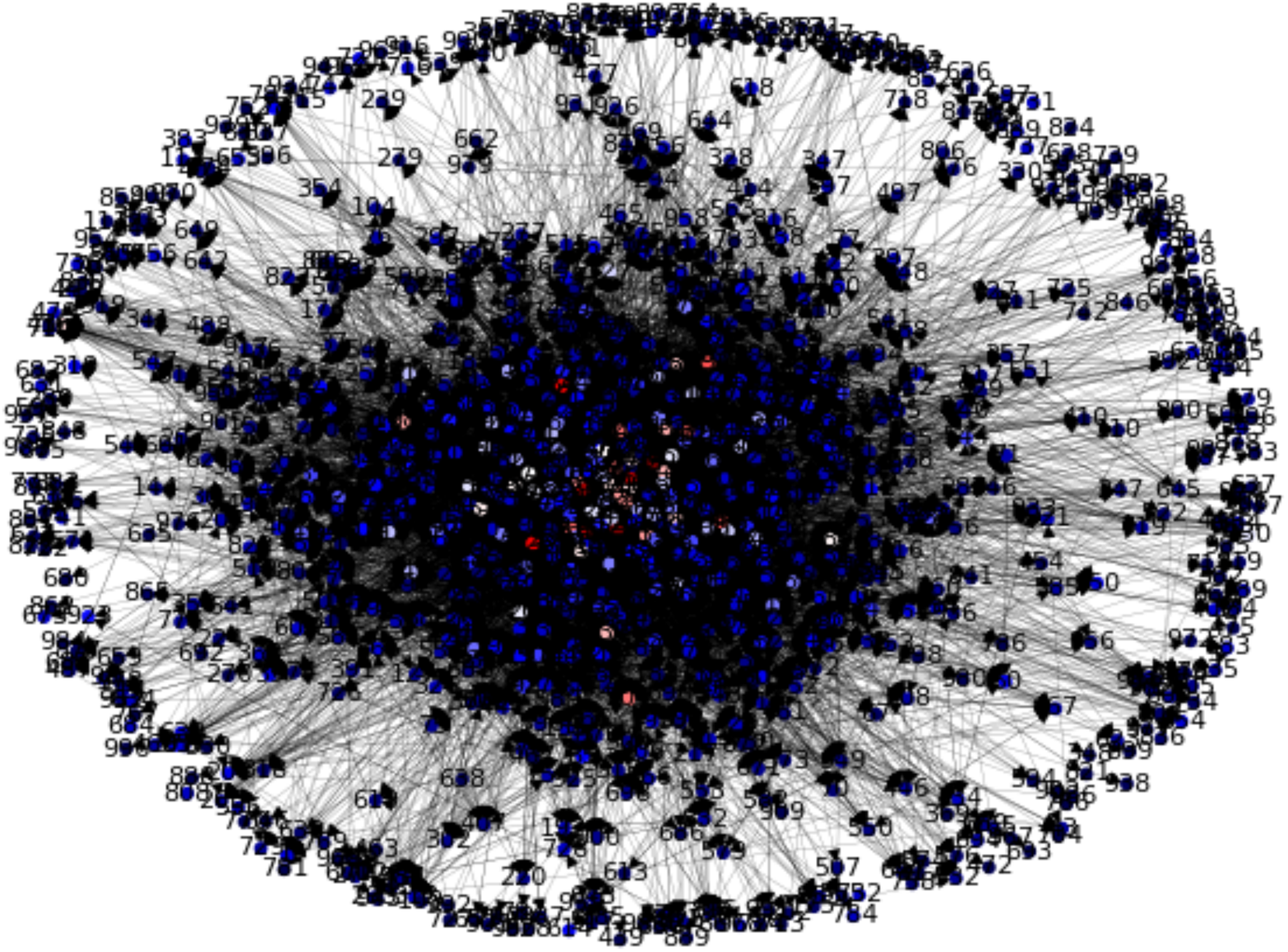
100 node dataset



500 node dataset



1000 node dataset

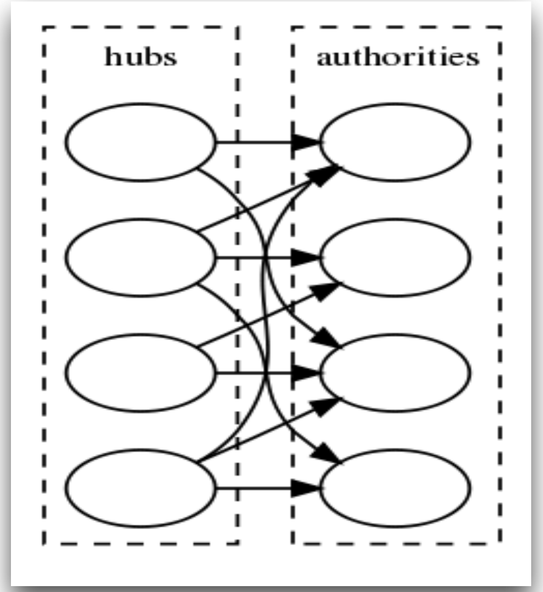


HITS algorithm

HITS algorithm

- HITS algorithm is a ranking method similar to PageRank, however because of the application of SVD in this method we are interested in this method.

- This method unlike the PageRank method that only considers the incoming edges, uses both incoming and outgoing edges to generate two ranking vectors one called the **authority vector** measuring the quality of incoming links similar to PageRank vector, and the other vector called the **hubness vector** measuring the quality of the outgoing links of a node.



Updating hubs and authorities

We denote the hubness vector with *h* and the authority vector with *a.*

Then updating in HITS algorithm is defined as h = Aa, *a* = A^T h, then we can

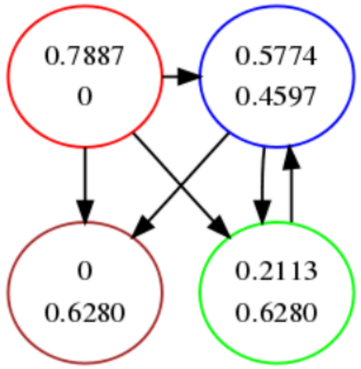
easily see:

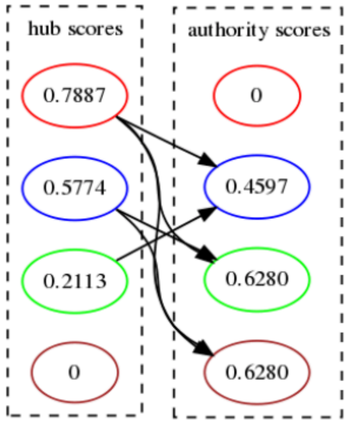
$$a = A^T Aa \text{ (I)}$$

$$h = A\, A^T h \text{ (II)}$$

From the above equations it follows that *a* is a right eigenvector of A^TA and *h* is a right eigenvector of AA^T. Now immediately the singular value decomposition of A comes to mind. When $A = U\Sigma V^T$, $A^TA = VSV^T$ and $AA^T = USU^T$, where *S* is the eigenvalue matrix of *A*. The first vectors of U and V are the first eigenvectors of AA^T and A^TA respectively which are the same as *h* and *a*.

Example





Adjacency matrix

	1	2	3	4
1	0	1	1	1
2	0	0	1	1
3	0	0	0	0
4	0	1	0	0

Eigenvalues

2.1753	1.1260	0	0

U

0.7886	0.2113	-0.5773	0
0.5773	-0.5773	0.5773	0
0	0	0	1
0.2113	0.7886	0.5773	0

V

0	0	0	1
0.4597	0.8880	0	0
0.6279	-0.3250	-0.7071	0
0.6279	-0.3250	0.7071	0

Infection spread modeling

- A contact network in which an infection spreads can be modeled using a general undirected graph*, since the spread of a virus can be modeled by a random walk in which nodes adjacent to an infected node could contract the infection by some probability ß, and each node *v* could heal by some probability c(v), and the network heals by the vector C, consisting of C(i)s.

- In this modeling a vector C is desired, such that the healing rate is highest possible, with minimal resources.

- Using the Cheeger ratio that measures bottlenecked-ness , and PageRank (any centrality); we can analyze the speed of spread and best distribution of antidotes (vector C).

References

1) Internet Mathematics Vol. 1, Deeper inside PageRank.
Amy N. Langville, Carl D. Meyer.

2) The anatomy of a large-scale hypertextual Web search engine. Sergey Brin, Lawrence Page. Computer science department, Stanford university. 1998.

3) MATHEMATICS BEHIND GOOGLE'S PAGERANK ALGORITHM. Brian Moore, 2018.

4) Distributing antidote using PageRank vectors. Fang Chung, Paul Horn, Alexander Tsiatas.

5) https://en.wikipedia.org/wiki/PageRank

6) https://en.wikipedia.org/wiki/Centrality

7) https://en.wikipedia.org/wiki/HITS_algorithm

8) https://computerscience.chemeketa.edu/cs160Reader/_static/pageRankApp/index.html (PageRank calculator).

9) https://snap.stanford.edu/data/ (Datasets are available here)