

Distributed and Adaptable Ranking Algorithm for Reputation and Relevance Extraction.

Josep M[†]. Pujol, Ramon Sangüesa[†], Jordi Delgado[†]

[†] Universitat Politècnica de Catalunya, LSI,
Jordi Girona Salgado, 1-3 08034 Barcelona, Spain.
{jmpujol, sanguesa, jdelgado}@lsi.upc.es

Abstract

We present *NodeRanking* a new mechanism for ranking the importance of nodes in a graph. It uses criteria similar to the ones used in well-known methods but it works by using local information and it is able to adapt dynamically to the graph topology. We show the quality of the algorithm by applying it to Social Networks and also to graphs that follow the Web's topological properties, the results obtained by *NodeRanking* will be compared with results obtained from *Pagerank* and *HITS* algorithms. We also show how the reputation of the members of a community can be extracted using *NodeRanking* and the community's Social Network.

Keywords: Reputation Mechanisms, Ranking Algorithms, Graph Algorithms, Reputation Measures, Web Mining, Social Networks, Web Intelligence.

1 Introduction

Ranking information on the web was done initially with a generic criterion based on the isolated content of the explored pages. Several techniques were used in the earlier versions of search engines. Variations on the frequency of terms and their relations were widely used. However, the rankings that engines based on this idea returned were not very coherent as users of the first well-known commercial search engines soon discovered. This started several research efforts that tried to find alternative methods based on criteria other than pure document content and layout in order to give better ratings and rankings. The structure of web references, i.e., the links that pointed to a given reference from other ones were a very good starting point to obtain better rankings mechanisms. Several algorithms are now in use that start from the principle that considering the World Wide Web as a graph, a node of this graph (i.e. a page or a document linked to it) is more important, that is, gets a better rating, if its pointed by more 'authoritative' nodes than other resources. Authority results from the number and

quality of the nodes that point to a resource. Basically this is the idea of the *HITS* and *Pagerank* algorithms [6, 10] that has been put to practical use with good results and wide acceptance. The key in this method is the transfer of confidence on the quality of a resource through a link. This idea is very intuitive and elegant. However, as the Web is also a web of people [1]. It would be interesting to explore if the relations of trust that evolve among people and are at the basis of one person referring to another one as a trustworthy source of knowledge or information are also of use for rating and ranking. Years of work on Social Network analysis [15] hint that the links existing between individuals in that type of net are trusted reputation indicators, that is, they are some type of indication on the trustworthiness of an individual. In that sense, the links in a social network are "stronger" than simple web references. The idea is, then to extract social networks from the world wide web in order to obtain an increase in the significance of the links. An algorithm working on the social network, giving that works on stronger links may give better results in terms of rating and ranking. We present *NodeRanking* an algorithm that is able to recognise the important nodes in a graph independently of the underlying topology of it using only local information, the knowledge about the whole graph is not necessary, each node knows only its relations. We test it in two different kind of networks: First on a network where links clearly are a representation of confidence: the social network of a real community. Secondly, in a network that had the same topological properties of the Web. Section 2, introduces the concept of social network; Section 3 describes some of the most reputed ranking algorithms as well as *NodeRanking* algorithm. Section 4 contains the experiments; and the last section closes with an overall discussion and hints on further research.

2 Social Networks

A *social network* [15] is a representation of the relationships existing within a community of people. Even within the same community several types of

social network can be built depending on the social relationship taken into account: kinship, acquaintanceship, friendship, mutual support, cooperation, similarity are typical criteria used in establishing the social relationship components of a community. The corresponding social networks are represented as graphs. See *Referralweb* [5] or *NetExpert* [12] for a multiagent system exploiting social networks on the web. The location of a given member of a community within a social network can be used to infer some properties about his or her degree of reputation in the community. Experts who are well-known and highly regarded by most other members of the community tend to be easily identified as highly connected nodes in the global graph [1,6,10]. This relational information could be a basis for a ranking mechanism instead of having to use ratings created on isolated information about an individual.

Our case study is based on the UPC Software Department social network. This social network is generated automatically by *NetExpert* [12] that is a part of a larger system called *Collaboratory* [13], which is multiagent system supporting a research community. It is basically a system that receives documents from users and has a set of recommender systems for delivering relevant information to users. The social network is built by analysing just the personal web pages of the community members.

Table 1 shows the most important characteristics of the undirected Social Network. To formalize the notion of a small world, Watts and Strogatz [16] define the clustering coefficient C , and the characteristic path length L . We take the diameter of a graph as well D . [2] Clustering coefficient is a real in the interval $[0..1]$.

	Social Network	Random Network
#nodes	139	139
#edges	394	394
C	0.7045	0.3774
L	3.6572	3.016
D	9	6

Table 1: Social Networks characteristics

Watts and Strogatz define a small world graph as one in which $L \approx L_{rand}$ and $C \gg C_{rand}$ where L_{rand} and C_{rand} are the characteristic path length and clustering coefficient of a random graph with the same number of nodes and edges. It is accepted that a small world graph should have $(C/C_{rand})/(L/L_{rand}) > 1$, our Social Network has $(C/C_{rand})/(L/L_{rand}) = 1.5394$.

The distribution of the node degree of our Social Network follows clearly an exponential distribution,

few nodes with very high degree and many with low degree. That confirms that the social Network is a small world in the Watts and Strogatz sense. In figure 1 the node degree histogram in a log scale is shown. Because of the few nodes that our Social Network has the accumulated frequency has been used instead of the frequency. Thus the fluctuation produced because of the lack of points can be reduced.

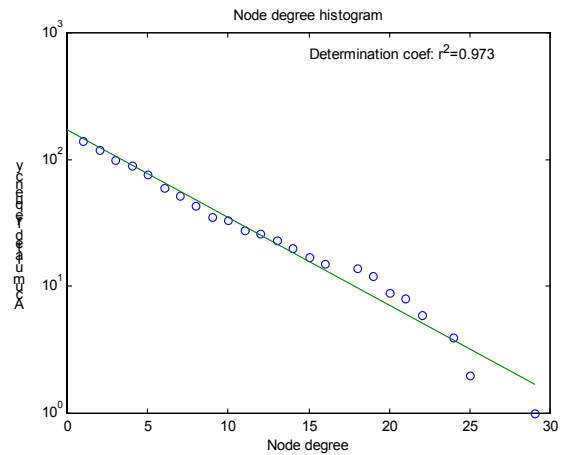


Figure 1. Node degree Histogram

Social network properties such as small world should be considered in order to improve knowledge extraction from the network and expertise location, because by exploiting these properties more efficient search [14] and propagation algorithms can be used, instead of traditional generic graph algorithms.

3 Ranking Algorithms

There are some algorithms that are able to extract the ranking by relevance of nodes using only the graph topology commonly named the links' structure [6,10]. There are also several improvements based on these algorithms or at least based on the underlying idea [8, 9]. Before present our algorithm, which is based on the concept of *authority* transfer through the edges, *Pagerank* [10] and *HITS* [6] will be explained briefly, these algorithms are very well known as a reference in the topology based ranking algorithms.

3.1 Overview of *Pagerank*.

The main idea of *Pagerank* [10] is that *good* nodes point or are pointed by another *good* nodes. So, the *authority* of a node depends of the nodes that point to it.

$$x^{(t+1)} = L^T x^t \quad (1)$$

Where L is the graph's adjacency matrix. L_{ij} contains the link from i to j . However the matrix LT is not used in *Pagerank*, that is based in a random walker strategy that can be modelled as a Markov process. Because of it *Pagerank* uses the matrix P , that is the adjacency matrix L , where all the rows sum to 1, thus a stochastic matrix is obtained. Moreover to avoid the *rank sink* [10] problem some probability of jumping to whatever node in the graph is introduced, this probability breaks the not truly increment of authority due to the cycles of the graph. The full stochastic matrix of transition probability is:

$$M = e \frac{1}{n} 11^T + (1-e)P \quad (2)$$

Where 1 is a vector of ones, n is the number of nodes and e is the jump probability. This jump probability has to be defined a priori and the recommended value by the authors is 0.15, further research works with values in the range [0.1..0.2].

The equilibrium distribution, that is the stationary state of the Markov chain defined by the transition matrix M can be obtained calculation the principal eigenvector of the matrix M^T .

3.2 Overview of HITS.

In the HITS algorithm [6], each page has both a hub score y_i and a authority score x_i . The main idea is that a good authority is pointed by many good hubs and a good hub points to many good authorities. As can be seen in next equations.

$$x_i = L^T y_j \quad (3)$$

$$y_i = Lx_i$$

The final scores of every node can be obtained through an iterative process.

$$\begin{aligned} x^{(t+1)} &= L^T Lx^{(t)} \\ y^{(t+1)} &= LL^T y^{(t)} \end{aligned} \quad (4)$$

Finally, the solution of authority, that is the stationary state of the equation 4, is the principal eigen vector of the $L^T L$ matrix. At the same manner the principal eigenvector of LL^T matrix contains the hub scores.

3.3 Our Proposal: NodeRanking.

NodeRanking is our proposal for creating a ranking of the graph nodes by means of the corresponding link topology.

The rating that *NodeRanking* creates is based on the idea that each node on the graph has an associated *degree of authority* that can be seen as an importance

measure. Initially, all nodes are assumed to have the same authority. After running *NodeRanking* the resulting authority measure is used to infer the reputation of a node within the graph, that is, the reputation of a member within his community.

Authority of a node, a , is calculated as a function of the total measure of authority present in the network and the authority of the nodes pointing to a . If a node is not referred by any other node in the network it is assigned a default authority value. Authority values are a positive values.

The algorithm for authority calculation is inspired in the ranking algorithms for web pages based on web topology [6,10]. The idea is to apply a similar reasoning about link topology in webpages to a the topology of a Social Network, i.e., the link topology of a directed graph. In a directed graph the edges have a direction, the out-edges of a node are the edges that start in this node, the out-nodes are the nodes that can be reached through out-edges.

The main idea of the *NodeRanking* algorithm is that each node has an authority and a part of its authority is propagated to the out-nodes via out-edges. The authority of a node depends on the authority of its in-nodes. Cycles in the graph can produce critical deadlocks in the calculations. The *NodeRanking* algorithm overcomes these problems and insures the convergence of calculations.

The NodeRanking Algorithm

NodeRanking follows a *random walker strategy* to explore the graph. It starts in a randomly selected node, and proceeds by selecting one of the nodes that can be reached through out-edges.

```

do
  n ← getNode(v)
do
  passAuthority(n)
  nnew ← getNextNode(n)
  n ← nnew
while (∃ nnew)
while (¬ converge())

```

Figure 2. *NodeRanking* algorithm

We have used some functions that require additional explanation.

getNode (): returns a randomly chosen node of the graph. The probability is uniform.

getNextNode(node x): returns one of the out-neighbours nodes of the node x . Each node has a set, that can be empty, of out-edges that points to other nodes. This function can return a null node to stop the path of the random walker by introducing some

elements of randomness. When *getNextNode()* returns a null node, in fact, the path is broken. There are two cases where the path is broken: 1) If the algorithm arrives at a node that has been already visited in the previous k steps. The algorithm keeps a window of k elements and breaks the path if a new node already exists in the window. 2) The walker evaluates the jumping probability whenever it reaches a node. This probability is a function of the connectivity of the node. The walker skips the node with a probability $\text{Pr}_{\text{jump}}(a)$.

$$\text{Pr}_{\text{jump}}(a) = \frac{1}{\#out - edges(a) + 1} \quad (5)$$

Nodes with fewer out-edges have a greater probability of breaking the path. This could be seen as a walker that gets bored because of the little range of choices. When *getNextNode()* returns the next node b to be visited from a . This node is selected with a probability that is calculated as a function of the weight of the edge between a and b , the corresponding density probability function is shown below.

$$\text{Pr}_{\text{choose}}(a \rightarrow b) = \frac{w(a \rightarrow b)}{\sum_{\forall \alpha \in out-nodes(a)} w(a \rightarrow \alpha)} \quad (6)$$

Where $w(a \rightarrow b)$ is the weight of the link connecting a and b .

passAuthority(node x): this function assigns part of the authority of node x to all the nodes that are pointed by x . The next equation shows the change of authority between a node x and a node y .

$$\Delta auth(y) = \left[\frac{\text{Pr}_{\text{choose}}(x \rightarrow y) auth(x)}{F_y} \right] \quad (7)$$

Where $auth(y)$ is the authority of the node y and F_y is a factor to maintain the value of authority within a limited range of values. Without this factor, values calculated with equation 7 would tend to infinity because the authority of a node gets bigger and bigger as the method proceeds. The factor F controls the growth of the authority and also eliminates the effects of randomness introduced by the asynchronicity in the authority updating. Each node has its own F factor that remembers the state of the total authority value of the graph the last time that the node was involved in a *passAuthority* function call. These factors depend on the authority value of all the nodes in the graph at the time that the authority of node y was updated. The

factors of more frequently visited nodes grow faster than the values of the less visited ones. The growth of these factor's values is monotonously increasing. Thus we can insure convergence towards a finite value. Without this factor nodes that have been last in the random selection would have an advantage over the other ones, because the graph is accumulating more and more authority as the algorithm proceeds. Factor F is initialized for every node as the sum of the authority of all nodes in the graph. The initial authority of a node has to be positive, and factor F has to be bigger or equal than 1.

converge(): this function can be evaluated anytime, it's a test to all the nodes of the graph. Each node remembers its last increment in the authority. The increment of authority tend to 0 because of the F factor, as can be seen in equation 7. The function converge test the state of each node, if the increment is minor than a given threshold, that is φ , the node will be considered as stationary. When all the nodes of the graph are stationary the algorithm ends. Actually the convergence does not test all the nodes because it is not very efficient, the event of becoming stationary is notified by the nodes themselves.

The algorithm's parameters that have been used along all the experiment are the following: $k=4$ and $\varphi=10^{-6}$.

3.4 Comparisons

All the algorithms explained previously implements the similar underlying idea but the running is different. However there are some differences that are significative.

Distribution Using Only Local Information

HITS and *Pagerank* are based on finding out the stationary state of a matrix, the variance-covariance matrix in the case of *HITS* and the transition probability matrix for *Pagerank*. In order to do it the adjacency matrix of the graph has to be known, moreover when the principal eigenvector is calculated following the technique named iterative product the vector, that contains the principal eigen vector, has to be normalized frequently, so all the the vector has to be known. *HITS* and *Pagerank* uses global information of the graph, it means that it is difficult to use directly these algorithms to rank huge graphs, think in a not very huge graph of 25000 nodes, the adjacency matrix has $6.25 \cdot 10^8$ positions. Because of the authority updating mechanism, that *Pagerank* and *HITS* uses, there is a synchronization in the process of authority transfer because the state of all the nodes has to be known in order to update the state of one of them. On the contrary *NodeRanking* uses only local information,

the nodes only have to know the nodes that he points and they are aware of his convergence to the stationary state. In order to retrieve the results of the ranking a centralization point is required, but even in this process the communication is one way from the nodes to the controller, so it is not necessary at all knowing the whole graph.

Pagerank and *Noderanking* are almost identical in the underlying idea, both of them follow the same random walker strategy. Actually the transition probability matrix of *Noderanking* can be defined as follows:

$$M = J \frac{1}{n} 11^T + (11^T - J)P \quad (8)$$

Where 1 is the vector of ones, n is the number of nodes, P is the adjacency matrix normalized by rows, J is the jumping probability matrix defined as a θ matrix where the diagonal contains the jumping probability of a node, i.e. J_{ii} contains node i jumping probability, that is defined in equation 5. The results obtained finding out the stationary state of the Markov chain defined by the transition probability matrix M , i.e. finding out the principal eigenvector of the matrix M^T , is equivalent to the results obtained by the original *Noderanking* algorithm. The advantage of the original *Noderanking* is that it is not necessary knowing the graph's adjacency matrix contrary from techniques based on transition probability matrix or variance-covariance matrix.

Dynamic Jumping Probability

As it can be seen in the equation 8 the matrix J contains the jumping probability of a node. Each node has different jumping probability that is calculated in the equation 5. So the jumping probability only depends on the local information. The average jumping probability depends on the distribution of nodes' out-degree, thus *NodeRanking* is able to adapt dynamically to graphs with different topologies because the jumping probability depend of the node's out-edges. It contrast with *Pagerank* where the jumping probability is not adaptative, mostly the jumping probability is fixed to 0.15 or values in the [0.1..0.2] range.

4 Experiments about ranking, reputation and relevance.

The experiments about *NodeRanking* as a ranking algorithm were splitted in two parts. The first was how can reputation of community members can be derived from the topology of the social network using 'web pages' ranking algorithms. An the second one was

about how *NodeRanking* adapts itself to rank well independently of the kind of graph.

4.1 Extracting Reputation from Social Networks.

The *NodeRanking* algorithm was applied to the social network of the experimental community formed by the members of our Software Department. Afterwards, a fragment of the community, 34 members, was selected randomly to become our test set. The rank obtained by means of *NodeRanking* was called $Rank_{NodeRanking}$. The ranking obtained by *NodeRanking* was compared against the results got by applying the *Pagerank* with the $\epsilon=0.15$ and also against the results got by applying the *HITS* algorithm.

The *Pagerank* and the *HITS* algorithms are used to work with graphs with no weighted edges. This kind of edges can be useful if they are available because the normalized weight of an edge contributes more information than the fact that an edge exists or not. The social network of our study has weighted edges, so two rankings instead of one were built for each algorithm, one of them was calculated without having the weights into account. They were $Rank_{PageRank}$ or $Rank_{HitsAuthority}$. The second ranking, $Rank_{PageRank(w)}$, was calculated by having them into account.

In order to validate the obtained rankings as a correct ranking measure they had to be compared against a real and accepted measure of importance with this type of network. The community behind the Social Network is a group of researches. It is important to test both algorithms on a Social Network because the alternative (to test them on the web) poses several problems. On the one hand, it is difficult to get a representative sample of resources from the web. On the other hand, it is difficult to analyze the quality of a resource on the net. Finally, the underlying idea of authority flux that is common to both algorithm is better represented in the links of a Social Network (in the sense of being a more significative measure) than in the links existing on the web. So, an appropriate way to establish a comparison was to resort to citation index impact values for each one of the 34 members calculated of the real community that appeared in the Social Network. A simple and recognized measure of 'authority' in scientific communities is the one obtained through the rankings of an independent scientific publication ranking agency. So, the citation indexes for each of the 34 randomly selected members of the Software Department were compared against the ratings that *NodeRanking* yielded ($Rank_{NodeRanking}$), against the ratings that *PageRank* yielded ($Rank_{PageRank}$ and ($Rank_{PageRank(w)}$) and also against the

ratings that *HITS* yielded ($Rank_{HITSAuth}$ and $Rank_{HITSAuth(w)}$).

CiteSeer [4] was used as a source for citation index values. The papers of each member of the community were retrieved from *CiteSeer* with the corresponding number of citations and self-citations for each them. We have built two rankings for the members of our test community. $Rank_{cite}$, sorts researchers by number of citations and $Rank_{cite-self}$, sorts researchers by number of citations without counting self-citations. These rankings can be considered as reference rankings, the closest ones to real reputation in the scientific community.

To compare the quality of the ratings obtained by *NodeRanking*, *Pagerank*, *HITS* and the reference rankings built using *Citeseer* the correlation coefficient between rankings was used as a similarity measure.

Table 2 shows the correlation values between reference rankings $Rank_{cite}$, $Rank_{cite-self}$ considered as the desired rankings, and the rest. The ranking $Rank_{NodeRanking}$ is the average of twenty executions of the algorithm, the mean and the standard deviation is given. That variability is due to the asynchronicity of the authority transfer process within *NodeRanking*. Independently of the reference ranking, there is always the same order among the generated rankings. $Rank_{NodeRanking}$ values are always a little better than $Rank_{PageRank}$ and $Rank_{PageRank(w)}$ values and these ones are better than the $Rank_{HITSAuth(w)}$ and $Rank_{HITSAuth}$. It is interesting to remark that the rankings obtained by *HITS* algorithm are better if the edge's weight is taken into account. However in any case the correlation is very strong, correlation coefficient goes from $[-1..1]$, where 0 means that there is no correlation, the correlation between two random rankings is close to 0. So, rankings with a correlation of 0.621 or 0.687 against the reference ranking can not be considered equivalent even though they are similar enough to be interesting.

Correlation coefficient	$Rank_{cite}$	$Rank_{cite-self}$
$Rank_{cite}$	1.0	0.983
$Rank_{cite-self}$	0.983	1.0
$Rank_{NodeRanking}$	0.687, s=8.6*10 ⁻⁴	0.621, s=0.011
$Rank_{PageRank(w)}$	0.535	0.486
$Rank_{PageRank}$	0.521	0.495
$Rank_{HITSAuth(w)}$	0.412	0.383
$Rank_{HITSAuth}$	0.342	0.323

Table 2. Correlation values between rankings

Figure 3 shows the correlation between $Rank_{cite}$ as a desired ranking and the rest of rankings

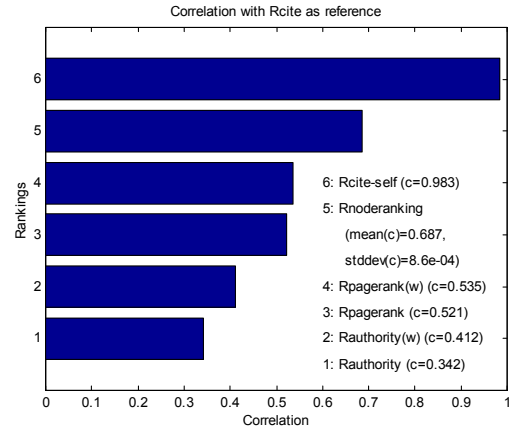


Figure 3. Correlation with $Rank_{cite}$ as reference

Figure 4 shows the correlation between $Rank_{cite-self}$ as a desired ranking and all the rest.

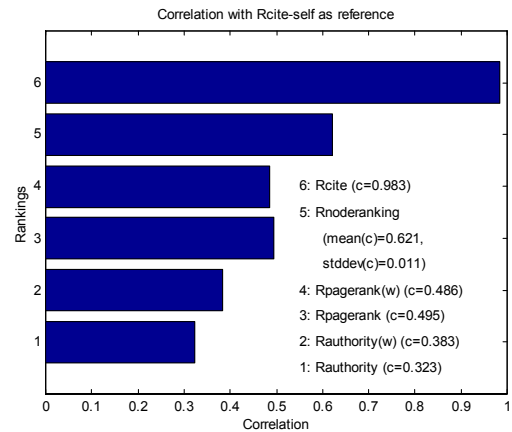


Figure 4. Correlation with $Rank_{cite-self}$ as reference

As it can be observed, the rankings obtained by *NodeRanking* are better than the *PageRank* ranking. Both algorithms follow the same random walker strategy with the difference of the authority updating mechanism, that is asynchronous in *NodeRanking* and synchronous in *Pagerank*. So the results should be similar, the reason of the best performance of *Noderanking* is due to its ability to adapt itself to the graph's topology.

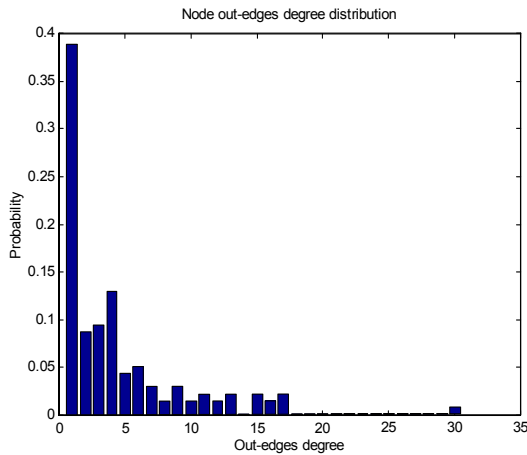


Figure 5. Social Network out-edges degree distribution

In figure 5 and also in figure 1 can be observed the distribution of node's out-edges degree, this distribution follows an exponential distribution that is characteristic in the graphs that are small worlds such as our social network is. The average jumping probability can be calculated using this distribution that will be called $d_{out-connectivity}$.

$$\overline{\Pr}_{jump} = \sum_i d_{out-connectivity}(i) \Pr_{jump}(i) \quad (9)$$

The average jumping probability of our case of study social network is 0.5314. So the *Noderanking* with its dynamical adjust of the jumping probability is able to adapt to different graph topologies, on the other hand *Pagerank* cannot rank social networks so well because of the exponential distribution of the connectivity, there is a lot of nodes with no out-edges or with very few out-edges, so a jumping probability of 0.15 is too small and *Pagerank* falls into the *rank sink* trouble. If *Pagerank* is set to work with a jumping probability of 0.5414 the results are equivalent between both algorithm, that is perfectly coherent. However the fact that *Noderanking* has only used local information is a good point.

4.2 Extracting Relevance from Web.

As it was mentioned before *NodeRanking* is able to adapts to the graph topology, in order to check it another experiment will be done comparing *NodeRanking* and *Pagerank*.

Now the graph will be a graph that has the topological properties of the Web. There is a lot of efforts in generating graphs that reproduce the properties that the Web has from its topology [7,3,11]. The

distribution of the out-edges degree in the Web decays as power-law instead the exponential distribution of social networks. There is some properties such as clustering coefficient, correlation between nodes that has been observed in the Web and other natural or artificial huge networks. There are several models that generates graphs that follows the Web topological properties. For our experiment it has been used the model proposed by Klem and Eguíluz [7]. This model create a graph that can be considered as a graph equivalent to the Web. The initial parameters are $m=8$, $a=2$ and $n=25000$. The model generates a 25000 nodes scale-free graph with a power-law distribution of connectivity with an exponent of 2.25, the same exponent that Internet has.

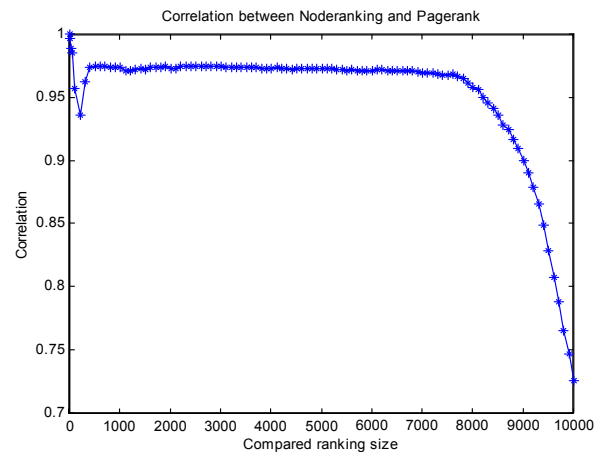


Figure 6. Correlation of rankings in the Klemm graph.

Figure 6 shows the correlation between the ranking obtained by *PageRank* and the ranking obtained by *NodeRanking* in the Klemm and Eguíluz graph, that has the Web topological properties.

The correlation of ranking obtained by *Pagerank* and *NodeRanking* in a web-based graph is very high as can be observed in figure 6. In the X axis there is the size of the compared ranking, the correlation of the first ten nodes is 0.9964, the correlation of the first one hundred is 0.9564, and the correlation of the first nine thousand is 0.8892. It can be concluded that the rankings obtained in huge graphs that have the Web's topological properties, and in the Web itself, are almost identically.

5 Discussion and further research

A new algorithm named *NodeRanking* has been devised to obtaining the ranking of graph's nodes by

relevance. The quality of the algorithm has been compared against the rankings yielded by *Pagerank* and *HITS* in two graphs that have different topological properties: one is a social Networks and the other is a web looks like graph. The values of the ratings calculated by the topological information of a community's social network has been used as a measure of each node in a graph. An experiment has been performed on a real community and the results compared against a well-known valid measure of reputation for that type of communities. The results seem to indicate that *NodeRanking* values are a good approximation of reputation measures if the graph is a social network where the links means acquaintance, another ranking algorithms such as *Pagerank* or *HITS* do not obtain such an acceptable approximation. Another experiment has been performed in a graph that follow the Web's topological properties are highly correlated with the ranking obtained by *Pagerank*.

NodeRanking is able to rank the nodes of a graph independently of its topology only by using local information. It does not need to know the entire graph to operate and it adapts dynamically to the analyzed graph topology.

This generic algorithm can be applied to rank web resources from the raw web in a similar fashion as *Pagerank* or *HITS* do and also to measure reputation of members within a knowledge community. Reputation is hot topic in multiagent systems research where it is used to reinforce cooperation between agent in response to the reputation of each one of them. We are currently using *NodeRanking* in a Multiagent System which supports a knowledge community, where it has shown a practical advantage with respect to other ways of measuring reputation. In effect, it does not require to have users continuously and explicitly issuing ratings [18], a method that is seen as a burden on users and eventually a reason for poor performance of collaborative systems. With our proposal a quite approximate reputation ranking can be calculated a priori without the typical and annoying feedback request used in collaborative systems. Other multiagent system methods that use social networks either don't use them for reputation measurement, as is the case of *Referralweb* [5] or still rely exclusively on rating feedback from users as [17] does. This last one has only been tested on a simulated community as opposed to a the test we carried on a real one (another example of this tests on simulated communities is [18]).

REFERENCES

- [1] Adamic, L.A., and Adar, E. Friends and neighbors on the web, 2000.
- [2] Adamic, L.A., The small world web. In S. Abiteboul and A.-M. Vercoustre, editors, Proc. 3rd European Conf. Research and Advanced Technology for Digital Libraries, ECDL, number 1696. Springer-Verlag, 1999.
- [3] Albert, A. and Barabasi, A.L. Topology of Evolving Networks: Local Events and Universality. Physical review Letters, vol. 85, pp. 5234—5237, 2000.
- [4] CiteSeer. Available on <http://www.citeseer.com/>.
- [5] Kautz, H., Selman, B., and Shah, M. The hidden web. AI Magazine, (18), 1997.
- [6] Kleinberg, J. Authoritative sources in a hyperlinked environment. Technical Report RJ 10076, IBM, 1997.
- [7] Klemm, K., and Eguiluz, V.M. Highly clustered scale-free networks. ArXiv:cond-mat/0107606, 2001.
- [8] Lempel, R. and Moran, S. The stochastic approach for link-structure analysis SALSA and the TCK effect. Proc. 9th WWW Conference, 2000.
- [9] Ng, A.Y., Zheng, A.X., and Jordan, M.I. Stable algorithms for link analysis. Proc. ACM Conf. On Research and Development in Information Retrieval (SIGIR'01), 2001.
- [10] Page, L., Brin, S., R. Motwani, T. Winograd, The PageRank citation ranking: Bringing order to the Web, submitted for publication.
- [11] Pastor-Satorras, R. And Vázquez, A., Vespignani, A. Dynamical and correlation properties of the Internet. ArXiv:cond-mat/0105161, 2001.
- [12] Sangüesa, R. and Pujol, J.M. NetExpert: A multiagent system for expertise location. Accepted to IJCAI'01 Workshop on Organizational Memories and Knowledge Management, 2001.
- [13] Vázquez, A., Barrio, I., Vázquez-Salceda, J., Pujol, J.M. and Sangüesa, R. An agent-based Collaboratory. Proceedings of ACAI2001 & EASS2001 Student Sessions, Prague, July 2001.
- [14] Walsh, T. Search in a small world. In Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99-Vol2), pages 1172--1177, S.F., July.
- [15] Wasserman, S. and Glaskiewics, J. Advances in Social Networks Analysis. Sage Publications, 1994.
- [16] Watts, D.J., and Strogatz, S.H., Collective dynamics of 'small-world' networks. Nature, (393), 1998.
- [17] Yu, B. and Singh, M.P, A Social Mechanism of Reputation Management in Electronic Communities. Proceedings of Fourth International Workshop on Cooperative Information Agents, pages 154-165, 2000.
- [18] Zacharias, G. and Maes, P., Trust Management Through Reputation Mechanisms, Applied Artificial Intelligence 14, pp 881-907, 2000.