2009 Ninth IEEE International Conference on Data Mining

Efficient Algorithm for Computing Link-based Similarity in Real World Networks

Yuanzhe Cai[1,2], Gao Cong[3], Xu Jia[1,2], Hongyan Liu[4], Jun He[1,2], Jiaheng Lu[1,2], Xiaoyong Du[1,2]

[1]Key Labs of Data Engineering and Knowledge Engineering, Ministry of Education, China
[2]Department of Computer Science, Renmin University of China, China
{yzcai, xujia, hejun, duyong}@ruc.edu.cn
jiahenglu@gmail.com
[3]Department of Computer Science, Aalborg University, Denmark
gaocong@cs.aau.dk
[4]Department of Management Science and Engineering, Tsinghua University, China
liuhy@ sem.tsinghua.edu.cn

Abstract—**Similarity calculation has many applications, such as information retrieval, and collaborative filtering, among many others. It has been shown that link-based similarity measure, such as SimRank, is very effective in characterizing the object similarities in networks, such as the Web, by exploiting the object-to-object relationship. Unfortunately, it is prohibitively expensive to compute the link-based similarity in a relatively large graph. In this paper, based on the observation that link-based similarity scores of real world graphs follow the power-law distribution, we propose a new approximate algorithm, namely Power-SimRank, with guaranteed error bound to efficiently compute link-based similarity measure. We also prove the convergence of the proposed algorithm. Extensive experiments conducted on real world datasets and synthetic datasets show that the proposed algorithm outperforms SimRank by four-five times in terms of efficiency while the error generated by the approximation is small.**

Keywords- Similarity Calculation, SimRank, Graph Mining

I. INTRODUCTION

It is a fundamental problem to compute similarity between objects in a network. The similarity calculation has a wide range of applications including similarity search, collaborative filtering for recommendation, clustering, etc. Many application domains can be modeled as networks. For examples, the Web graph (to find and recommend similar webpages), the paper citation graph (to find similar papers or related authors), the query-click graphs (to recommend similar queries and webpages), and social network and collaborative tagging websites (to recommend friends and objects, such as photos).

Link-based similarity [1] has been proposed to describe the similarity between objects in graphs by exploiting the object-to-object relationship. It has been shown that link-based similarity measures produce more consistent results with human judgment than conventional similarity measures based on content (e.g. text for webpages) similarity, in many application domains [2].

The basic idea of link-based similarity is that two objects are similar if they are related to similar objects in the networks. One of the most well known link-based similarity measures is `SimRank` [1] and many applications are developed, e.g. [3-7] based on `SimRank` and its extensions. Intuitively, similar to `PageRank` algorithm, `SimRank` is also based on the concept of "random surfers", and accordingly it is computed iteratively. Although it has been shown that `SimRank`, as a link-based similarity measure, is very effective in measuring the similarity of objects in networks, it is unfortunately computationally expensive. Its time complexity is $O(kn^4)$, where k is the number of iterations required for the `SimRank` algorithm to converge, and n is the number of objects in a network.

In this paper, we address the problem of efficiently computing link-based similarity. Our solution is based on an important observation that link-based similarity values follow power law distribution. It has been well studied that degrees in many real world graphs follow the power-law distribution (such graphs are called scale free graphs), such as the World Wide Web [9-10], metabolic network [11], telephone call graph[12], paper citation graph [13], etc. However the power law distribution for link-based similarity is less observed and studied. We observe that it exists as a fundamental and essential property for real world graphs (Section 3 will give detailed analysis).

A salient feature of power-law distribution of link-based similarity scores is that majority of similarity scores between objects in real world graphs have very small values while only a small part of similarity scores are large. Based on this feature, we propose approximate algorithm, namely `Power-SimRank`, for computing `SimRank`. We prove that the approximate algorithm has guaranteed error bounds, and that the proposed algorithm will converge.

Our contributions:

- We offer a detailed study on an important principle in real world graphs: power-law distribution for link-based similarity scores.
- Based on the characteristics of power-law distribution in similarity scores, we develop an approximate algorithm with guaranteed error bounds, namely `Power-SimRank` to extend `SimRank`. We also prove that the proposed algorithm converges by theoretical analysis.
- Extensive experiments are conducted to evaluate the efficiency of the proposed algorithm and also the accuracy estimation of the proposed algorithm. Experimental results show that the

Power-SimRank algorithm is capable of outperforming SimRank by 4-5 times in terms of runtime, while the error generated by the approximation is small.

The rest of the paper is organized as follows: Section 2 introduces the SimRank algorithm. Section 3 presents the power-law distribution property of link-based similarity in real world graphs. Section 4 presents the proposed Power-SimRank algorithm. Experimental results are reported in Section 5. We cover related work in Section 6. Finally, Section 7 concludes our work.

II. Overview on SimRank

SimRank [1] is a method of measuring linkage-based similarity between objects in a graph that models the object-to-object relationships for a particular domain. The intuition behind SimRank score is that two objects are similar if they link to the similar objects. This intuition also indicates that SimRank calculation needs to be recursive.

We proceed to present the formula to compute SimRank. Given a graph $G(V, E)$ consisting of a set of nodes V and a set of links E, the SimRank similarity between objects a and b, denoted as $S(a, b)$, is computed recursively as follows:

$$S(a,b)=\begin{cases}1 & (a=b)\\ \dfrac{c}{|I(a)||I(b)|}\sum_{i=1}^{|I(a)|}\sum_{j=1}^{|I(b)|}S(I_i(a),I_j(b)) & (a\neq b)\end{cases} \qquad \textbf{(1)}$$

,where c is a constant decay factor, 0< c <1; $I(a)$ is the set of neighbor nodes of a and $I_i(a)$ is the i^{th} neighbor node of a. $|I(a)|$ is the number of neighbors of node a. In case of $I(a)$ or $I(b)$ being an empty set, $S(a, b)$ is specially defined as zero.

A solution to the SimRank equation (1) can be reached by iteration to a fixed-point. For each iteration k, let $S_k(.,.)$ be an iteration similarity function and $S_k(a, b)$ be the iterative similarity score of pair (a, b) on iteration k^{th}. The iteration process is started with S_0 as follow.

$$S_0(a,b)=\begin{cases}0 & (\text{if } a\neq b)\\ 1 & (\text{if } a=b)\end{cases} \qquad \textbf{(2)}$$

To calculate $S_{k+1}(a, b)$ from $S_k(a, b)$, we have the following equation:

$$S_{k+1}(a,b)=\frac{c}{|I(a)||I(b)|}\sum_{i=1}^{|I(a)|}\sum_{j=1}^{|I(b)|}S_k(I_i(a),I_j(b)) \qquad \textbf{(3)}$$

In equation (3), $1/|I(a)|$ is a single step probability of walking from node a to a node in $I(a)$. Therefore we can use probability transformation matrix T [21] to capture the single step probability in a Markov Chain. Thus, SimRank algorithm can be described by matrix calculation.

$$S_0=E \qquad \textbf{(4)}$$

,where E is an identity matrix.

$$S_k(a,b)=c\cdot\sum_{i=1}^{|I(a)|}\sum_{j=1}^{|I(b)|}T_{aI_i(a)}\cdot T_{bI_j(b)}\cdot S_{k-1}(I_i(a),I_j(b)) \qquad \textbf{(5)}$$

Although the convergence of iterative SimRank algorithm can be guaranteed in theory, practical computation uses a tolerance factor ε to control the number of iterations such that a finite number of iterations are performed. It is recommend to set ε = 0.001, the same as in PageRank [21]. Specifically, the ending condition of the iteration is as follows:

$$max(|S_k(a,b) - S_{k-1}(a,b)| / |S_{k-1}(a,b)|) \leqslant \varepsilon \qquad \textbf{(6)}$$

It says that the iteration stops if the maximal change rate of similarity value between two iterations for all node pairs is smaller than the threshold ε.

III. Power-Law of SimRank Scores

In this section, we study SimRank score of node pairs in scale free graphs. We discretize similarity scores into bins of size 0.002; each bin (s-0.002, s] is represented by s. Let f_s be the number of node pairs whose similarity scores follow in bin s. Figure 1 (a) shows the frequency f_s versus the similarity score s in the log-log scale on different graphs. A striking obervation is that the plots can be approximated well by the linear regression. For example, the correlation coefficient is 2.17 for Cornell dataset. This shows that SimRank scores follow power-law distribution.

Power-Law Distribution of SimRank scores
The frequency f_s is a function of SimRank *score s: $f_s = \Phi s^{\zeta}$, where ζ and Φ are two constant parameters.*

Definition 1: We define SimRank exponent ζ to be the slope of the plot of frequency of SimRank scores versus SimRank score in log-log scale.

The second striking observation is that the values of SimRank exponent ζ remains relatively stable across the datasets. Generally, the exponent parameter ζ is in the range [-1.8, -2.2] and details are given in Table 1.

We next analyze the fraction of node pairs whose SimRank scores exceed s over all node pairs, and the fraction of the sum of their SimRank scores over the sum of scores of all pairs.

Theorem 1: *Let $P(s) = \int_s^{+\infty} f_s ds / \int_{s_{\min}}^{+\infty} f_s ds$ be the fraction of the nodes pairs whose* SimRank *score exceeds s, and $W(s) = \int_s^{+\infty} sf_s ds / \int_{s_{\min}}^{+\infty} sf_s ds$, be the fraction of total* SimRank *score for these pairs. ζ >2. Then, we have $W=P^{(\zeta-2)/(\zeta-1)}$* **(7)**

***Proof*:**

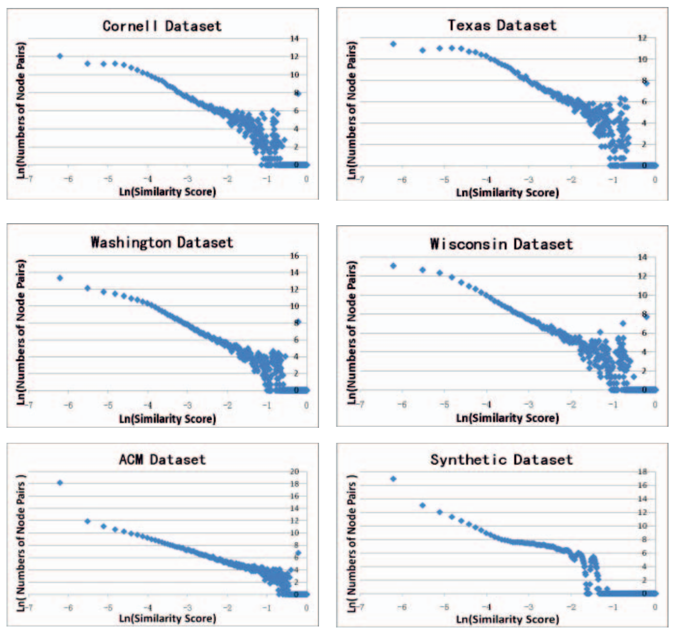
$$P(s)=\int_s^{+\infty} f_s ds/\int_{s_{\min}}^{+\infty} f_s ds = \int_s^{+\infty}\phi s^{\zeta}\, ds/\int_{s_{\min}}^{+\infty}\phi s^{\zeta}\, ds = (s/s_{min})^{-\zeta+1}$$

$$W(s)=\int_s^{+\infty} sf_s ds/\int_{s_{\min}}^{+\infty} sf_s ds = \int_s^{+\infty}\phi s^{\zeta+1}\, ds/\int_{s_{\min}}^{+\infty}\phi s^{\zeta+1}\, ds = (s/s_{min})^{-\zeta+2}$$

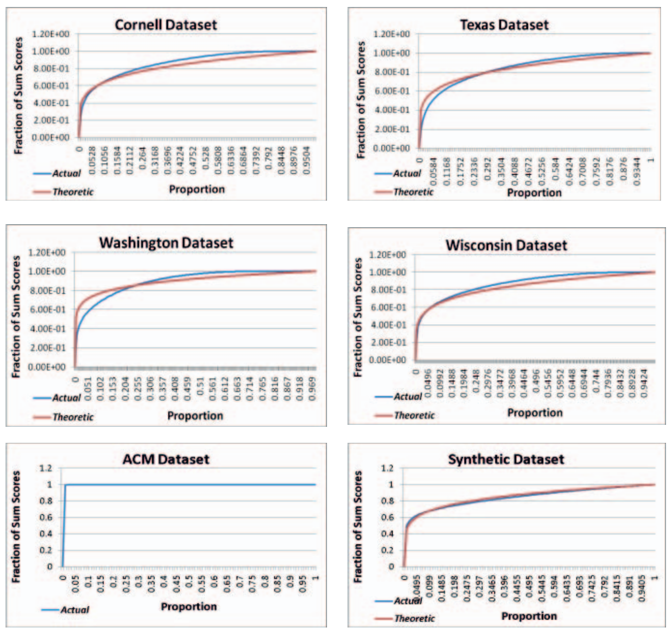
Thus, $W = P^{(\zeta-2)/(\zeta-1)}$.■

According to theroem 1, the P-W curve is concave downwards with a steep increase. This indicates that majority of the SimRank scores are very small and a small portion of the nodes pairs take a large fraction of SimRank scores in real world networks.

We apply theorem 1 on our datasets to plot P-W curves (the fraction of SimRank scores vs. the fraction of node pairs). Figure 1(b) shows the P-W curves according to Theorem 1 and the real values on different datasets. We can see that the two curves match very well. Take Cornell dataset as an example, we can see that 20% node pairs with the largest SimRank scores hold about 78% of the sum of SimRank scores over all node pairs. We note that the curve of ACM dataset appears to be different. This is



(a)



(b)

Figure 1. SimRank scores for Power-Law distribution on real world datasets: (a) Log-log plot of frequency f versus the similarity score s. (b) Fraction Proportion: the actual line describes the fraction W of the sum of SimRank scores held by fraction P of pairs and the theoretic line is drawn by following Theorem 1. The detail information on the datasets is shown in Table 1.

because ACM dataset is extremely sparse (Table 1), the similarity scores of ACM dataset are more smaller than other datasets and a large number of similarity scores are 0. The parameter ζ of ACM dataset is inapplicable to evaluate faction proportion. However, for the extremely sparse graph of ACM dataset, 2% node pairs with the largest SimRank scores hold all of SimRank scores. Hence, in general, in real world networks, fewer than 20% node pairs with the largest SimRank scores hold the 80% SimRank scores.

TABLE I. PARAMETER STATISTICS OF DATASETS

Dataset	Vertices#(n)	Edges#(e)	β	δ_1	ζ	δ_2
Cornell	867	2667	-2.77	0.101	-2.17	0.002
Texas	827	2691	-2.65	0.079	-2.16	0.003
Washington	1205	3299	-2.54	0.068	-2.10	0.002
Wisconsin	1263	5305	-2.23	0.046	-2.16	0.001
ACM	8983	7855	-3.04	0.050	-1.80	0.002
Synthetic	5000	19980	-3.93	0.062	-2.15	0.005

Note: β is the slope of the degree distribution graph in log-log scale. ζ is the slope of SimRank score distribution graph in log-log scale. δ_1 and δ_2 are standard deviation for β and ζ respectively. The parameters β and ζ are calculated by MLE [8].

IV. POWER-SIMRANK ALGORITHM

The observations on power-law distribution for similarity scores show that a large portion of SimRank score is owned by a small percentage of nodes pairs for real world networks and the majority of similarity scores is very small. This indicates that the time of similarity computation can be greatly reduced if we can save the computation used to compute the low similarity scores, which costs most of the computation. We conjecture that we can save computation time of SimRank if we do not recalculate P proportion of node pairs after a certain number of iterations, while we can guarantee that the errors from stopping the recalculation is within certain bounds.

Based on this idea, we propose a new algorithm Power-SimRank that is capable of computing approximate SimRank with guaranteed error bounds. Algorithm 1 outlines the Power-SimRank algorithm. It takes in five arguments. The first three arguments inherit from the original SimRank algorithm: the decay factor c gives the rate of decay as similarity flows across edges in a graph; transfer probability matrix T is explained in section 2.1; and tolerance factor ε is to control the number of iterations as discussed in section 2.1. The other two parameters are iteration threshold r and fraction of nodes pairs P. Intuitively, after iteration r, we will stop recalculating the P portion of node pairs with the lowest similarity score. As to be shown, they will decide the theoretical error bounds of the approximated SimRank scores, and thus can be set in terms of the accuracy requirements of different applications.

The algorithm first initializes variables (lines 1-3). Each element in matrix flagmatrix indicates whether or not the similarity of the corresponding node pair needs be calculated. If flagmatrix[a,b] = 0, pair(a, b) will need to be calculated iteratively. If flagmatrix[a,b] = 1, we will stop the iterative computation for the pair(a, b), and we call the pair (a, b) is locked. In line(4), The algorithm will stop if the ending condition in Equation (6) is satisfied. The algorithm then uses Equation (5) to compute the similarity score if node pair pair(a, b) is not set to 1 (lines 7-10). In the r^{th} iteration, if any node pair(a,b) is included by the P fraction (i.e. its similarity score is in the lowest P percentage), we will set flagmatrix[a,b] to 1 and will not recalculate its SimRank score in the subsequent iterations (lines 11-15). LockPairsNodes is used for this function.

Algorithm 1. Power-SimRank

```
Input: Decay Factor c, Transfer Probability Matrix T, Tolerance Factor ε, Iteration Threshold r, Fraction of Nodes Pairs P
Output: Similarity Matrix s
1:  k←1;
2:  S_0←identity matrix;
3:  flagmatrix ←0; //0:node pair need calculate. 1:do not need
4:  while(max(|S_k(a, b) - S_{k-1}(a, b)| / |S_{k-1}(a, b)|) > ε )
5:      k←k+1;
6:      S_{k-1}←S_k;
7:      for each element S_k(a, b)
8:          if ( flagmatrix(a, b)= 0 )
9:              $S_k(a,b) = c \cdot \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} T_{aI_i(a)} \cdot T_{bI_j(b)} \cdot S_{k-1}(I_i(a), I_j(b))$
10:     end for
11:     if ( k = r )
12:         for each element S_k(a, b)
13:             LockPairsNodes(S_k, P); // Set Flagmatrix
14:         end for
15:     end if
16: end while
17: return S_k
```

Although the worst time and space complexity of Power-SimRank is the same as SimRank, it is much more efficient in practical computation since it needs fewer iterations and in each iteration the computation cost is cheaper than SimRank.

Theorem 2 (Convergence): *The* Power-SimRank *similarity S (a, b) for any node pair (a, b) will converge to a fixed value.*

Proof: See the full version of this paper[14].

We also proceed to derive the error bound for Power-SimRank algorithm. The parameters *r* and *P* for Power-SimRank are significant to analyze the accuracy of SimRank.

Theorem 3 (Accuracy Rule). *At the iteration r, the similarity of P fraction of node pairs is locked (no further calculation) by the* Power-SimRank. *Thus, for every two nodes a and b, the difference between the* SimRank *and* Power-SimRank *similarities*, i.e. *the theoretical error bounds, is given as follows:*

Case 1: *If pair (a, b) does not need further calculation beyond the iteration r*th*, S(a, b)-PS(a, b)≤ c^{r+1}.*

Case 2:*If pair(a, b) needs to calculate beyond the iteration r*th*,*

$$avg(S(a,b) - PS(a,b)) \le P^{\frac{2-\zeta}{1-\zeta}} c^{r+2} \frac{1}{1-(c(1-P^{\frac{2-\zeta}{1-\zeta}}))} \quad \textbf{(8)}$$

Proof: See the full version of this paper[14].

V.EMPIRICAL STUDY

We report a summary of empirical study on the performance of our methods. Section 5.1 presents the experimental setting, section 5.2 reports the efficiency and section 5.3 reports the accuracy of the similarity measures for real world applications.

A. Experimental Setting

Datasets: Our experiments use the datasets in Table 1.

Cornell, Texas, Washington, and Wisconsin Datasets: They consist of web pages crawled from computer departments of four universities from CMU datasets [15]. Web pages are manually divided into seven classes, student, faculty, staff, department, course, project and others. Each webpage becomes a graph node.

ACM Dataset: The dataset consists of 8968 papers crawled from Section B in ACM CCS [16], which is a credible subject classification system for Computer Science. Each paper is a graph node. According to ACM CCS 1998, these papers has been classified for six classes, such as "B.1control structures and microprogramming".

Synthetic Dataset: We use Barabasi Graph Generator [17] to generate graphs such that their node degree follows the power-law distribution as follows: 5 initial nodes are created. Then upon each iteration of the generator, a new node is added and 2 edges are created from this new node to other nodes with the probability distribution $P(i) = d_i / \sum d_i$, where d_i is the degree of node *i*. This process continues until the desired size of the graph is reached. We generate a number of graphs with the number of nodes from 500 to 5000 and use V_xE_y to denote the generated graph with *x* nodes and *y* edges.

All our experiments are conducted on a PC with a 3.0GHz Intel Core 2 Duo Processor, 2GB memory, running windows XP Professional. All algorithms are implemented in Java.

B. Efficiency of the proposed algorithm

Table 2 gives the runtime of the the proposed methods Power-SimRank, and the orginal SimRank. For Power-SimRank, we use default parameter *r* = 6 and *P* = 20%. We can see that Power-SimRank speeds up SimRank by nearly four times on all data. The main reason for the speed-up is that Power-SimRank needs much less time after certain (*r*) interations than SimRank and our methods need less iteration. Note that the computational cost for all algorithms depends on both the time for each step of iteration and the number of iterations.

TABLE II. TOTAL TIME(S) VS. ALGORITHM

Dataset	SimRank	Power-SimRank
Cornell	162.0s	**33.3s**
Wisconsin	602.7s	**132.3s**
Texas	147.9s	**32.1s**
Washington	453.1s	**93.8s**
ACM	191441s	**47599s**
$V_{5000}E_{2000}$	37946s	**10508s**

To have a better understanding on the speed-up, Figure 4 shows the time cost for each iteration of different algorithms on all the data. We can see that from the first to the 6th iteration, the time cost of each iteration of Power-SimRank is almost identical to that of SimRank, but at subsequent iterations the time cost of our algorithm drops dramatically. After the 6th iteration, 80% of the nodes with low similarity scores are locked in the proposed methods. The subsequent iterations will only compute similarity score for unlocked node pairs (nearly 20% node pairs according to power law distribution), and thus the time cost of our algorithm drops very quickly. Note that the proposed algorithm need fewer iterations to converge as shown in Figure 2.

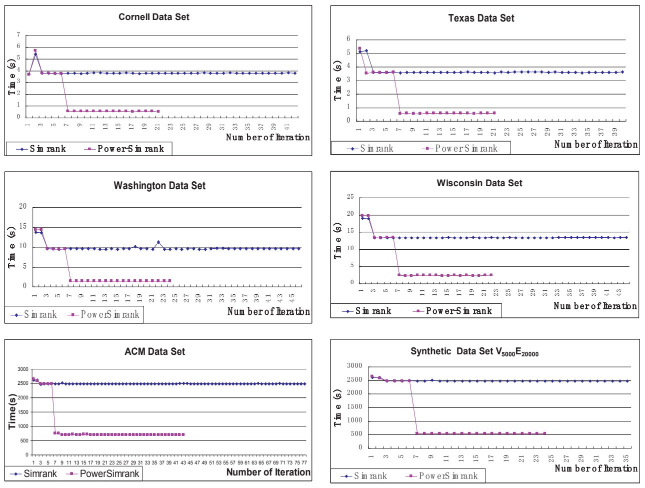


Figure 2. Number of iteration vs. Time (seconds)

To study the scalability of the proposed methods, we run experiments on synthetic graphs with different number of nodes. As shown in Figure 3, the runtime increases with the number of nodes for all methods. However, `Power-SimRank` scales much better than `SimRank`, and runs nearly four times faster than `SimRank` algorithm.

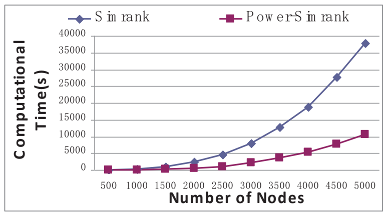


Figure 3. The number of nodes vs. Time(seconds)

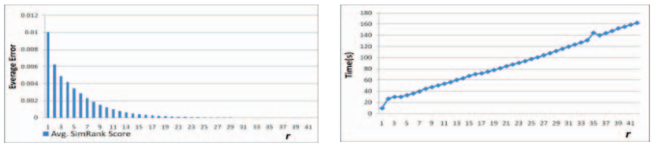
C. *Accuracy of the proposed algorithm*

We evaluate the accuracy in two ways. First, we compute the average error of similarity scores of our algorithm compared with `SimRank`. Second, we evaluate the accuracy of using the similarity score from different methods for a clustering application.

1) *Average Error*

Two parameters, *r* and *P*, affect the accuracy of similarity values directly. We will evaluate the effect of the two parameters on the performance. We observe qualitatively similar tendency for `Power-SimRank` on all datasets. Due to space limitation, we only report the result of `Power-SimRank` on the Cornell dataset.

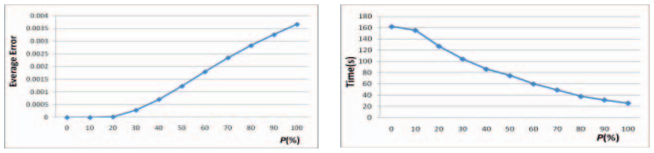
In the first experiment, we fix the parameter *P* to 0.8 and vary *r* from 1 to 41. Figure 4(a) shows that with the increment of *r*, the average error (the error of a node pair is the difference between Power-SimRank and SimRank for the node pair) of Power-SimRank drop. In figure 4(a), the line Pc^n describes the trend of Ave SimRank score. With the iteration increase, the line accord with Pc^n. After 6th iteration, we also notice that the real average error is extremely small. In Figure 4(b), with the increase of *r*, the runtime of Power-SimRank increases.



(a) Average Error (b) Runtime

Figure 4. The effect of *r* in `Power-SimRank` on performance ($P = 0.8$)

In addition, we fix *r* at 6 and vary *P* from 0% to 100%. In figure 5, with the increase of *P*, the average error increases slowly from 0 to 0.0035 and the runtime drops quickly form 160s to 20s as expected.



(a) Average Error (b) Runtime

Figure 5. The effect of *P* in `Power-SimRank` on performance ($r = 6$)

2) *Accuracy for the real world application*

Note that the data we use has class labels that can be used as the ground truth for clustering. We follow the evaluation method in [3], which uses PAM [22], a k-medoids clustering approach, to cluster objects based on similarity score. We do the clustering using the similarities calculated by different algorithms. Precision [23] is used to measure the quality of clusters that are generated using different similarity metrics, and thus the quality of similarity scores:

$$precision = \frac{C_{r_1} + C_{r_2} + \ldots + C_{r_n}}{C_{t_1} + C_{t_2} + \ldots + C_{t_n}} \quad \textbf{(9)}$$

Where C_{r_i} is the number of nodes which have the right label in i^{th} cluster and C_{t_i} is the total number of nodes in i^{th} cluster. Following [3], for each similarity calculation method, we run PAM 50 times for each data and report the results of the run with the best precision. We use the default parameter *P* =0.8 and *r*=6 in this experiment.

In figure 6, it shows the accuracy of clusters generated using the four similarity measure. We can see that the precision using `Power-SimRank` similarity is slightly lower than that use `SimRank` similarity. However, the loss of the accuracy of `Power-SimRank` is small.

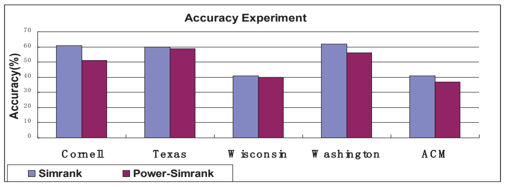


Figure 6. Accuracy of clustering using different similarity calculation methods on real world datasets

VI. RELATED WORK

The early research work for similarity calculation based on link analysis focuses on the citation patterns of scientific papers, based on measures `co-citation` [18] and `co-coupling` [19]. `co-citation` means if two documents are often cited together by other documents, they may be about the same topic. co-coupling means that if two papers cite many papers in common, they may be on the same topic. Amsler et al. [20] fuse bibliographic co-coupling and co-citation measures to determine the similarity between documents. These methods compute the similarity by considering only the immediate neighbors. In contrast, SimRank [1] considers the entire relationship graph to compute the similarity between two nodes. But this method suffers from high computational cost, $O(kn^4)$ in the worst case. There are some recent methods proposed to improve the performance of SimRank. Jeh and Widom [1] also propose an algorithm called Pruning-SimRank [1], which computes similarities by a small scope of relationship graph around two nodes. However, the accuracy of this method is not very good since it ignores much information. Lizorkin et al. [7] present a technique to estimate the accuracy of computing SimRank iteratively and some optimization techniques that improve the performance of the iterative algorithm. The techniques are orthogonal to our method and can be combined with our method. Fogoras et al. [6] presents a general framework of Monte Carlo similarity search algorithm that recomputes an index database of random fingerprints, and at query time, similarities are estimated from the fingerprints. Cai et al. [5] propose an Adaptive-SimRank algorithm based on the observation that the convergence rates of different object pairs are different when SimRank is used to compute the similarities. Xi et al [4] propose SimFusion measure that can effectively integrate relationships from multiple sources to measure the similarity of data objects by iteratively computing over the Unified Relationship Matrix (URM). All these techniques for link-based similarity do not consider power-law distribution of similarity scores that exists in real world graphs, as we observe in this paper. To our knowledge, the power-law distribution of SimRank scores is only observed previously by Yin et al [3] and they propose an efficient and accurate approach for linkage-based clustering algorithm LinkClus. However, Yin et al. do not consider parameter estimation for the power-law distribution, which is essential to study the power-law distribution. Moreover, we utilize the power-law distribution in a different way from Yin et al. Yin et al. propose a hierarchical structure, SimTree, to represent the similarity score between objects based on the power law distribution of SimRank scores and develop linkage-based clustering algorithm LinkClus. In contrast, we utilize the property of power-law distribution to reduce the number of iterations and computational cost of each iteration.

VII. Conclusion

In this paper, we find that link-based similarity scores follow the power-law distribution in real world graphs, and also estimate the parameters of the distribution. Based on this observation, we propose the Power-SimRank algorithm to improve the performance of SimRank. Moreover, we prove the convergence of the proposed algorithm and analyze the error bounds of the proposed algorithm. Experimental results show that the proposed algorithm runs four-five times faster than SimRank, and the approximation error is small.

Acknowledgments.

This work was supported in part by the National Nature Science Foundation of China under Grant No. 70871068, 70621061, 70890083, 60873017 and 863 High-tech Project (No. 2009AA01Z133).

References

[1] G. Jeh, J. Widom, “SimRank: a measure of structural-context similarity”, SIGKDD, pp. 538-543, 2002.

[2] H. Roinestad, and A. Vespignani. “Algorithmic computation and approximation of semantic similarity”, WWW, pp. 431-456 2006.

[3] X.X Yin, J.W Han, P.S. Yu, “LinkClus: efficient clustering via heterogeneous semantic links”, VLDB, pp. 427-438, 2006

[4] W.S. Xi, E. A. Fox, W. G. Fan, B. Y. Zhang, Z. Chen, J. Yan, D. Zhuang, "SimFusion: measuring similarity using unified relationship matrix", SIGIR, pp. 130-137, 2005

[5] Y.Z. Cai, H.Y. Liu, J. He, X.Y. Du, X. Jia, “An adaptive method for efficient similarity calculation”, DASFAA, pp. 339-353, 2009

[6] D. Fogaras, B. Racz, “Scaling link-base similarity search”, WWW, pp. 641-650, 2005

[7] L. Dmitry, V. Pavel, G. Maxim, T. Denis, “Accuracy Estimate and Optimization Techniques for SimRank Computation”, VLDB, pp. 422-433, 2008

[8] M. Goldstein, S. Morris, G. Yen, “Parameter estimation for power-law distributions by maximum likelihood methods”, Theoretical Physics, vol. 58(2), pp. 167-17, 2007

[9] A. Medina, I. Matta, J. Byers, “ On the Origin of Power Laws in Internet Topologies”, SIGCOMM, vol. 1(2), pp. 18-28, 2000

[10] M. Faloutsos, Faloutsos, P., Faloutsos, C. “On Power-Law Relationships of the Internet Topology”, SIGCOMM, pp. 251-262, 1999

[11] H. Jeong, S. Mason, A. L. Barabasi, and Z. N. Oltvai, “Lethality and centrality in protein networks”, Nature, vol. 41, pp. 41-42 2001

[12] W. Aiello, F. Chung,and L. Y. Lu, “Random evolution of massive graphs”, STOC, pp. 97-122, 2000

[13] S. Redner, “How popular is your paper? An empirical study of the citation distribution”, Euro. Phys. , pp. 131-134, 1998

[14] Y.Z. Cai, G. Cong, X. Jia, H.Y. Liu, J. He, J.H. Lu, X.Y. Du, “ Efficient Algorithms for Computing Link-based Similarity in Real World Networks”, Technical Report, 2009

[15] CMU Dataset, http://www.cs.cmu.edu/afs/cs/project/theo-20/www/data/

[16] ACM dataset, http://www.acm.org/

[17] A. Barabasi, R. Albert, “Emergence of Scaling in Random Networks”, Science, vol. 286(5439), pp. 509-512, 1999

[18] H. Small, “Co-citation in the scientific literature: A new measure of the relationship between two documents”, Journal of the American Society for Information Science, pp. 265-269, 1973.

[19] M.M. Kessler, “Bibliographic coupling between scientific papers”, American Documentation, pp. 10-25, 1963.

[20] R. Amsler, “Applications of citation-based automatic classification”, Technique Report, http://openlibrary.org/b/OL19540128M/Applications_of_citation-based_automatic_classification, 1972.

[21] A. N. Langville, C. D. Meyer, “Deeper inside PageRank”, Internet Mathematics, vol. 1(3), pp. 335-380, 2005

[22] J.W. Han, M. Kamber, “Data mining concepts and techniques”, Morgan Kaufmann Publishers, 2001

[23] D. S. Modha and W. S. Spangler, “Feature Weighting in k-Means Clustering”, Machine Learning, vol. 52(3), pp. 217-237, 2001