Comprehensively Computing Link-based Similarities by Building A Random Surfer Graph

Mingxi Zhang[1], Xifeng Yan[2], Wei Wang[3]
[1]University of Shanghai for Science and Technology, Shanghai, China
[2]University of California, Santa Barbara, California, USA
[3]Fudan University, Shanghai, China
mingxizhang10@fudan.edu.cn,xyan@cs.ucsb.edu,weiwang1@fudan.edu.cn

ABSTRACT

Link-based similarity computation arises in many real applications, including web search, clustering and recommender system. Lots of similarity measures are devoted recently, but there is one undesirable drawback, called “path missing” issue, *i.e.*, the paths between objects are not fully considered for similarity computation. For example, SimRank considers only in-coming paths of equal length from a common “center” object, and a large portion of other paths are fully neglected. A comprehensive measure can be modeled by tallying all the possible paths between objects, but a large number of traverses would be required for these paths to fetch the similarities, which might increase the computational difficulty. In this paper, we propose a comprehensive similarity measure, namely RG-SimRank (Random surfer Graph-based SimRank), which resolves the “path missing” issue with inheriting the philosophy of SimRank. We build a random surfer graph by allowing the surfer to stay at current object, go to other objects against in-links or along out-links. RG-SimRank adopts SimRank to compute similarities in random surfer graph instead of the original network, which has a same form of SimRank and hence inherits the optimization techniques on similarity computation. We prove that RG-SimRank considers all the possible paths of any direction and any length. And it provides a general solution to assess similarities, under which lots of existing similarity measures become its special cases. Other similarity measures besides SimRank can also be enhanced similarly using random surfer graph. Extensive experiments on real datasets demonstrate the performance of the proposed approach.

CCS CONCEPTS

• **Information systems** → **Similarity measures**.

KEYWORDS

similarity measure, random surfer graph, comprehensiveness

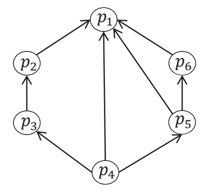
ACM Reference Format:

Mingxi Zhang, Xifeng Yan, Wei Wang. 2021. Comprehensively Computing Link-based Similarities by Building A Random Surfer Graph. In *Proceedings of the 30th ACM Int’l Conf. on Information and Knowledge Management (CIKM ’21), November 1–5, 2021, Virtual Event, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482329

1 INTRODUCTION

Link-based similarity computation focuses on assessing the similarities between objects in a given network, which arises in many real applications, including web search, clustering and recommender system. Link-based similarity measures exploit the object-to-object relationships expressed in terms of links for assessing object similarities. Compared to content- or text-based similarity measures, which treat each object as a bag of items, link-based similarity measures could produce systematically better correlation with human judgements [26].

Despite of the merits of link-based similarity measures, there exists a drawback, called “path missing” issue, *i.e.*, the paths between objects are not fully considered for similarity computation. For example, SimRank [10] considers only in-coming paths of equal length from a common “center” object, and thus a large portion of other paths are fully neglected. In real applications, *e.g.*, literature search in citation network [42, 49] and page ranking in web network [45], neglect of the paths might lead to incomprehensive results. Consider an example of citation network in Figure 1, where each node represents an object of paper type, and each edge represents a citation. We retrieve the similar papers for query p_2 by SimRank and reverse SimRank (rvs-SimRank) using the decay factor $c = 0.8$. The 2-tuples in the rankings represent the returned objects with similarity scores. Due to lack of consideration for the paths beyond in-coming direction, some similar objects are neglected by SimRank. For example, p_5 is neglected, though it is connected with p_2 through path $p_2 \rightarrow p_1 \leftarrow p_5$, since the directions from p_2 and p_5 to “center” object p_1 are out-going, and SimRank considers only in-coming paths like $p_2 \leftarrow p_3 \leftarrow p_4 \rightarrow p_5 \rightarrow p_6$ from “center” object p_4. It is rather counter-intuitive since papers cite similar papers should also be similar intuitively [49]. Similarly, p_4 is neglected though there is a path $p_2 \leftarrow p_3 \leftarrow p_4$ of mixed directions with “center” object p_3. In fact, p_4 should be similar to p_2 since both “cite” and “cited by” relationships can convey similar topics [42]. Rvs-SimRank returns these objects, but p_1 is neglected since it considers only out-going direction. There are also some similar objects neglected due to lack of consideration for the paths beyond equal length. For example, SimRank still neglects p_5 though there is an in-coming path $p_2 \leftarrow p_3 \leftarrow p_4 \rightarrow p_5$, since the path lengths from p_2 and p_5 to “center” object p_4 are not equal, and similarly p_3 is neglected by rvs-SimRank. Practically, the paths of different lengths also convey similar topics [45], which contribute to similarity scores.



Rank	SimRank	rvs-SimRank
1	(p_6, 0.640)	(p_6, 0.800)
2	(p_1, 0.160)	(p_5, 0.400)
3	N/A	(p_4,0.267)
4	N/A	N/A
5	N/A	N/A

Figure 1: Retrieved similar objects for a given query p_2 by SimRank and rvs-SimRank in citation network

The "path missing" issue roots not only in SimRank but also in lots of other similarity measures. For example, [5, 19, 42, 49] exploit both in- and out-links for computing similarities between objects, but only the paths of equal length are considered; and [6, 18, 45, 47] utilize the paths of different lengths, but they consider only either in- or out-link direction rather than both directions.

In this paper, we study the "path missing" issue of similarity computation. It is challenging to integrate the paths of any direction and any length. A straightforward approach is to tally all the possible paths between objects, but a large number of traverses would be required for these paths to fetch the similarities, which might increase the computational difficulty. While significant efforts are devoted to optimizing SimRank computation recently (*e.g.*, [17, 21, 23, 25, 31, 43, 44]), the "path missing" issue has attracted little attention. Recently, some similarity measures are extended from SimRank, such as P-Rank [49] and C-Rank [42], which try to utilize the information beyond in-coming paths with inheriting the beauty of SimRank philosophy. Benefit from the optimization techniques of SimRank, the similarities can be efficiently computed, but they are practically insufficient due to lack of enough consideration for different path lengths. Some metrics, such as SimRank* [45] and PageSim [18], integrate paths of different lengths for similarity computation, but the links of different directions are not fully utilized. And they are not in a SimRank form, which cannot be speeded up by the optimization techniques of SimRank. Therefore, it is non-trivial to seek a solution to comprehensively compute similarities with inheriting the beauty of existing studies especially the philosophy of SimRank so that lots of previous studies can be still applicable to similarity computation.

Fortunately, we observe that the paths between objects can be fully integrated by building a random surfer graph rather than rebuild a new similarity model. Specifically, the random surfer graph is derived by allowing the surfers to stay at current object, go to other objects against in-link direction or along out-link direction, so that the paths of any direction and any length can be considered. Based on random surfer graph, existing measures can be adopted directly to compute similarities without destroying the original philosophy. Our main contributions are as follows.

(1) We propose RG-SimRank (Random surfer Graph-based SimRank) and justify its comprehensiveness. We build a random surfer graph and then obtain RG-SimRank by applying SimRank to random surfer graph instead of the original network. RG-SimRank is proved to be a natural way of traversing all possible paths that are neglected by existing measures.
(2) We show that the RG-SimRank has a same form of SimRank, which inherits the basic philosophy of SimRank, and lots of existing optimization techniques are applicable to the similarity computation.
(3) A general solution is provided to assess similarities, under which lots of existing similarity measures can be considered as the special cases of RG-SimRank. And other similarity measures besides SimRank can also be enhanced using random surfer graph.
(4) Extensive experiments on real datasets demonstrate the performance of RG-SimRank through comparing with the state-of-the-art similarity measures.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 defines the "path missing" issue of existing similarity measures. Section 4 introduces random surfer graph, and proposes RG-SimRank with its comprehensiveness, inheritance and derivatives. Experimental studies are reported in section 5. Section 6 concludes the paper.

2 RELATED WORK

Co-citation [32] and Bibliographic coupling [14] are two noteworthy measures in bibliometrics field. Co-citation measures the similarity between papers based on the common papers which cite both of them, while Bibliographic coupling defines similarity as the number of their cited papers. These two measures utilize only the direct connections, and the indirect connections are fully neglected.

SimRank [10] defines similarity based on the intuition that "two objects are similar if they are referenced by similar objects", which is consistent to our basic understandings. As a widely-accepted measure, SimRank gained tremendous popularity in many vibrant communities, *e.g.*, similarity search [48], clustering [41] and recommender system [29]. SimRank does not suffer from any field restrictions and can be applied to any domain with object-to-object relationships. However, an unsatisfactory trait is that SimRank considers only the paths of in-link direction and equal length, and lots of other paths are neglected.

There are also some similarity measures of this kind. MatchSim [20] defines the similarity between objects as the average similarity of the maximum matching between their neighbors. RoleSim [12] utilizes a generalized Jaccard coefficients to ensure automorphic equivalence for SimRank. SimRank++ [3] compensates for the cardinality of in-neighbor matching by adding an evidence weight. SimFusion [39] computes similarities iteratively over a unified relationship matrix (URM). PathSim [33] and HeteSim [30] define similarities by counting the paths between objects via the instances of meta path. NetSim [46] employs SimRank to measure the similarities between attributes over attribute network. These measures consider only either in- or out-link direction, and the path lengths are limited to equal.

To utilize the paths of different lengths, some similarity measures are devoted. PageSim [18] measures similarities by counting the paths of any length based on PageRank score propagation. HeteRank [47] integrates the paths of any length over a general relationship matrix. SimRank* [45] resolves the "zero-similarity" issue by revising SimRank with tallying the paths of any length. SNS [24] computes similarities by learning node embeddings from neighbor information and local subgraphs. CSE [6] measures similarities based on direct links and common neighbors. These measures consider the paths consisting of links of either in- or out-link direction rather than both directions.

Some similarity measures compute similarities by exploiting both in- and out-links. Amsler [2] integrates Co-citation [32] and Bibliographic coupling [14] into similarity computation. ENS [19] utilizes both in- and out-link relationships to extend existing models. [5] extends SimRank by utilizing both in- and out-link relationships. P-Rank [49] enriches SimRank by jointly encoding both in- and out-links into structural similarity computation. C-Rank [42] extends SimRank by ignoring link directions for measuring similarity. However, these measures consider only the paths of equal length.

Random walk-based measures define the similarity as the random walk distance between objects. [36] suggests Random Walk with Restart (RWR) for measuring proximities, which can be considered an extension of Personalized PageRank (PPR) [11]. [27] proposes a random-walk method by exploiting PPR to compute similarities. [15] extends RWR by integrating independent and sensible coefficients. [52] computes similarities based on RWR in an attribute augmented graph. These measures consider the hitting probability from a node to another via the paths consisting of only either in- or out-links.

3 THE "PATH MISSING" ISSUE

We first give the formal definition of "path missing" issue, and then show that this issue is rooted in SimRank and some other metrics. For a given network, denoted as a directed graph $G = (V, E)$ with V denoting object set and E denoting link set, we shall abuse the following notions. (1) An *in-link path* between objects $a \in V$ and $b \in V$ is a walk of length $l_1 + l_2$, denoted by $a = u_0 \leftarrow u_1 \leftarrow \ldots \leftarrow u_{l_1} = x = v_{l_2} \rightarrow v_{l_2-1} \rightarrow \ldots \rightarrow v_0 = b$, starting from a, taking l_1 steps against the directions of in-links $u_{i-1} \leftarrow u_i$ for $i = 1, 2, \ldots, l_1$ and l_2 steps along the directions of out-links $v_j \rightarrow v_{j-1}$ for $j = 1, 2, \ldots, l_2$, and finally arriving at b, where x is the *in-link "center"*. (2) An *out-link path* between a and b is a walk of length $l_1 + l_2$, denoted by $a = u_0 \rightarrow u_1 \rightarrow \ldots \rightarrow u_{l_1} = x = v_{l_2} \leftarrow v_{l_2-1} \leftarrow \ldots \leftarrow v_0 = b$, starting from a, taking l_1 steps along the direction of $u_{i-1} \rightarrow u_i$ for $i = 1, 2, \ldots, l_1$ and l_2 steps against the direction of $v_j \leftarrow v_{j-1}$ for $j = 1, 2, \ldots, l_2$, and finally arriving at b, where x is the *out-link "center"*. (3) A *mixed-link path* between a and b is a path with mixed directions, denoted by $a = u_0 \dashrightarrow u_1 \dashrightarrow \ldots \dashrightarrow u_{l_1} = x = v_{l_2} \dashleftarrow v_{l_2-1} \dashleftarrow \ldots \dashleftarrow v_0 = b$, where $u_{i-1} \dashrightarrow u_i$ is a mixed-link, *i.e.*, in- or out-link, from u_{i-1} to u_i for $i = 1, 2, \ldots, l_1$, and x is the *mixed-link "center"*. A path is *symmetric* (in terms of length) if $l_1 = l_2$; otherwise, it is *dissymmetric*. A path of *any* length can be symmetric or dissymmetric.

Definition 3.1. When assessing similarity score between a and b, a similarity metric is comprehensive within path length l if it considers all the mixed-link paths of any length $l_1 + l_2$ for $l_1, l_2 \leq l$, starting from a and finally arriving at b; otherwise, it is a metric with "*path missing*" issue.

The "path missing" issue of SimRank is shown as:

Theorem 3.2. *At iteration l, SimRank considers only the symmetric in-link paths of length $l_1 + l_2$ for $l_1 = l_2 \leq l$.*

Proof. The SimRank score between a and b is defined as $S(a, b) = 1$ if $a = b$, otherwise

$$S(a, b) = \frac{c}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} S(I_i(a), I_j(b)) \tag{1}$$

where c is the decay factor between 0 and 1, $I(a)$ is the in-neighbor set of a, and $I_i(a)$ is the i-th neighbor of a. Let $R_l(a, b)$ be the computational SimRank score between a and b at iteration l, which is started with $R_0(a, b) = 1$ if $a \neq b$, otherwise $R_0(a, b) = 0$. When $l \neq 0$, $R_l(a, b)$ is defined as $R_l(a, b) = 1$ if $a = b$, and otherwise

$$R_l(a, b) = \frac{c}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} R_{l-1}(I_i(a), I_j(b)) \tag{2}$$

which is further derived as

$$R_l(a, b) = c \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} P_{a,I_i(a)} P_{b,I_j(b)} R_{l-1}(I_i(a), I_j(b)) \tag{3}$$

where $P_{a,I_i(a)}$ is the transition probability from a to $I_i(a)$ against in-link direction, defined as $P_{a,I_i(a)} = \frac{1}{|I(a)|}$ if $I(a) \neq \emptyset$ and otherwise $P_{a,I_i(a)} = 0$. We assume $R_l(a, b)$ considers only the symmetric in-link paths of length $l_1 + l_2$ between a and b like $a = u_0 \leftarrow u_1 \ldots \leftarrow u_{l_1} = x = v_{l_2} \rightarrow v_{l_2-1} \rightarrow \ldots \rightarrow v_0 = b$ for $l_1 = l_2 \leq l$. By Equation (3), we have

$$R_{l+1}(a, b) = c \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} P_{a,I_i(a)} P_{b,I_j(b)} R_l(I_i(a), I_j(b))$$

By the assumption, we get that $P_{a,I_i(a)} P_{b,I_j(b)} R_l(I_i(a), I_j(b))$ contains only in-link paths like $a \leftarrow I_i(a) = u_0 \leftarrow u_1 \leftarrow \ldots \leftarrow u_{l_1} = x = v_{l_2} \rightarrow v_{l_2-1} \rightarrow \ldots \rightarrow v_1 \rightarrow v_0 = I_j(b) \rightarrow b$, also denoted as $a = u'_0 \leftarrow I_i(a) = u'_1 \leftarrow \ldots \leftarrow u'_{l'_1} = x = v'_{l'_2} \rightarrow v'_{l'_2-1} \rightarrow \ldots \rightarrow v'_1 = I_j(b) \rightarrow v'_0 = b$ for $l'_1 = l'_2 \leq l + 1$. Replace l'_1 and l'_2 by l_1 and l_2 respectively, we have that $R_{l+1}(a, b)$ considers only symmetric in-link paths of length $l_1 + l_2$ between a and b for $l_1 = l_2 \leq l + 1$, which finishes the induction. □

The "path missing" issue of SimRank might lead to incomprehensive results. Specifically, the SimRank score $S(a, b) = 0$ if there does not exist any symmetric in-link path between a and b, as shown in Figure 1. More importantly, SimRank still "partially miss" all the contributions of dissymmetric in-link paths, such as path $a = p_2 \leftarrow p_3 \leftarrow x = p_4 \rightarrow p_1 = b$, and symmetric out-link paths, such as $a = p_2 \rightarrow p_1 = x \leftarrow p_6 = b$, even if $S(a, b) \neq 0$. Rvs-SimRank considers symmetric out-link paths, but the in-link paths are neglected, such as symmetric in-link path $a = p_2 \leftarrow p_3 \leftarrow p_4 = x \rightarrow p_5 \rightarrow p_6 = b$ and dissymmetric in-link path $a = p_2 \leftarrow p_3 \leftarrow p_4 = x \rightarrow p_5 = b$.

Some variants of SimRank try to remedy this issue. P-Rank [49] is derived from SimRank for utilizing both in- and out-link directions, but not all the symmetric mixed-link paths are utilized since the directions in the symmetric position of the path are limited to be same, *i.e.*, in a mixed path $a = u_0 \dashrightarrow u_1 \dashrightarrow \ldots \dashrightarrow u_{l_1} = x = v_{l_1} \dashleftarrow v_{l_1-1} \dashleftarrow \ldots \dashleftarrow v_0 = b$, the directions of $u_i \dashrightarrow u_j$ and $v_i \dashleftarrow v_j$ should be same. Compared to P-Rank, C-Rank [42] defines similarities by fully integrating the symmetric mixed-link paths without restrictions on link directions. However, both of them do not consider the dissymmetric paths. SimRank* [45] considers the in-link paths of any length, like $a = u_0 \leftarrow u_1 \leftarrow \ldots \leftarrow u_{l_1} = x = v_{l_2} \rightarrow v_{l_2-1} \rightarrow \ldots \rightarrow v_0 = b$ of length $l_1 + l_2 = 2i$ for $i = 1, 2, \ldots, l$, but the paths beyond in-link direction are neglected. Besides, SimRank* fails to utilize the optimization techniques of SimRank since it is presented by a different model.

RWR [36] and PPR [11] and their reverse forms consider the paths consisting of links of one single direction, and such paths can be considered as the special cases of in-link paths, out-link paths or mixed-link paths practically. For example, the path $a = u_0 \rightarrow u_1 \rightarrow \ldots \rightarrow u_i = b$ can be considered as an in-link path of length $0 + i$ with in-link "center" a, out link-path of length $i + 0$ with out-link "center" b, or mixed-link path $a = u_0 \rightarrow u_1 \rightarrow \ldots \rightarrow u_{i_1} = x \rightarrow u_{i_1+1} \rightarrow \ldots \rightarrow u_{i_1+i_2} = u_i = b$ of length $i_1 + i_2$ with "center" node x for any $i_1 + i_2 = i$. From this perspective, the mixed-link paths beyond above are fully neglected.

4 RG-SIMRANK

For comprehensively computing similarities, we need to consider the mixed-link paths of any length, which requires a large number of traverses for these paths to fetch the similarity scores. In this section, we propose RG-SimRank to remedy the "path missing" issue. We first build a random surfer graph for the given network, and then derive RG-SimRank by applying SimRank directly to random surfer graph instead of the original network, so that the philosophy of SimRank can be inherited and the previous optimization techniques on SimRank would be still applicable to the similarity computation.

4.1 Building random surfer graph

Consider a surfer randomly walking in a network, we allow it to stay at current object, go to other objects against the direction of in-links or along the direction of out-links at each step. And the corresponding transitions are called *stay transition*, *in-link transition* and *out-link transition* respectively. The random walk process is described by a random surfer model, defined as:

Definition 4.1 (Random surfer model). A random surfer starting from an object can either stay at the current object with *stay transition probability* $\gamma \in [0, 1]$ or leave with probability $1 - \gamma$ at each step. And then, when the random surfer decides leaving current object, it can either go to its in-neighbors against the directions of in-links with *in-link transition probability* $\lambda \in [0, 1]$ or go to its out-neighbors along the directions of out-links with *out-link transition probability* $1 - \lambda$.

Accordingly, the random surfer graph is defined as:

Definition 4.2 (Random surfer graph). The random surfer graph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = V$, and $\mathcal{E}$ consists of the available links a surfer can go under the random surfer model by starting from any object in G.

The matrix of in-link transition probabilities over the random surfer graph $\mathcal{G}$ is denoted by $\mathcal{P}$ with entry $\mathcal{P}_{a,x}$ denoting the in-link transition probability from object a to object $x \in V$, which is derived as $\mathcal{P}_{a,x} = \gamma$ if $a = x$, otherwise

$$\mathcal{P}_{a,x} = (1 - \gamma)\lambda P'_{a,x} + (1 - \gamma)(1 - \lambda)P_{a,x} \tag{4}$$

where $P'_{a,x}$ is the transition probability from a to x along out-link direction, defined as $P'_{a,x} = \frac{1}{|O(a)|}$ if $O(a) \neq \emptyset$ and otherwise $P'_{a,x} = 0$, and $O(a)$ is the out-neighbor set of a. And then, matrix $\mathcal{P}$ is derived as

$$\mathcal{P} = \gamma I + (1 - \gamma)(\lambda P' + (1 - \lambda)P) \tag{5}$$

where P and P' are the matrix form of $P_{*,*}$ and $P'_{*,*}$ respectively, I is an identify matrix of $n \times n$, and n is the object number. When $I(a) \neq \emptyset \wedge O(a) \neq \emptyset$, we get $\sum_{\forall x} \mathcal{P}_{a,x} = 1$; and otherwise, we get $\sum_{\forall x} \mathcal{P}_{a,x} \leq 1$, so we need to normalize each row vector by $\mathcal{P}_{a,*} = \frac{\mathcal{P}_{a,*}}{\sum_{\forall x} \mathcal{P}_{a,x}}$ to ensure $\sum_{\forall x} \mathcal{P}_{a,x} = 1$.

In previous studies, there are several concepts related to random surfers graph. One typical concept is the random surfer-pair model (*e.g.*, [9, 10, 28, 34, 49]) which allows surfers randomly walk on a given network. However, they consider only the walks of either different directions or different lengths. Furthermore, by these definitions, we need to rebuild a new model to integrate the paths, so the beauty of existing approaches cannot be inherited. Another concept is the random graph modeling (*e.g.*, [1, 4, 7, 8]) that mainly studies on the graph with randomly generated edges, which is rather different to random surfer graph and obviously not applicable to the "path missing" issue. Compared to above approaches, we integrate the mixed-link paths by building a random surfer graph uniformly so that existing similarity measures can be directly applied to the graph, which consequently inherits the philosophy of existing metrics without any destruction.

4.2 RG-SimRank model

4.2.1 RG-SimRank equation. RG-SimRank is derived by applying SimRank to random surfer graph instead of the original network. Formally, the similarity between a and b is defined as $S(a, b) = 1$ if $a = b$, otherwise

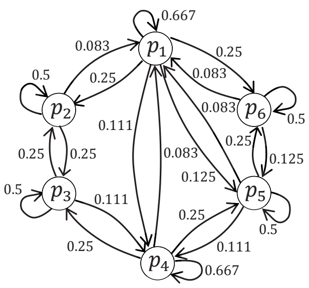
$$S(a, b) = c \sum_{i=1}^{|\mathcal{I}(a)|} \sum_{j=1}^{|\mathcal{I}(b)|} \mathcal{P}_{a,\mathcal{I}_i(a)} \mathcal{P}_{b,\mathcal{I}_j(b)} S(\mathcal{I}_i(a), \mathcal{I}_j(b)) \tag{6}$$

where $\mathcal{I}(a)$ is the in-neighbor set of a in $\mathcal{G}$ and $\mathcal{I}_i(a)$ is the i-th neighbor of a. RG-SimRank is computed in an iterative manner as SimRank. At iteration l, the similarity between a and b is denoted by $R_l(a, b)$, which is initialized as $R_0(a, b) = 1$ if $a = b$, otherwise $R_0(a, b) = 0$; and at $l = 1, 2, \ldots, R_l(a, b)$ is computed as $R_l(a, b) = 1$ if $a = b$, otherwise

$$R_l(a, b) = c \sum_{i=1}^{|\mathcal{I}(a)|} \sum_{j=1}^{|\mathcal{I}(b)|} \mathcal{P}_{a,\mathcal{I}_i(a)} \mathcal{P}_{b,\mathcal{I}_j(b)} R_{l-1}(\mathcal{I}_i(a), \mathcal{I}_j(b)) \tag{7}$$

where parameters λ and γ can be typically set as 0.5 as demonstrated in our experiments, and parameter c can be set as 0.8 according to the literature [10].

4.2.2 Comprehensiveness of RG-SimRank. By building random surfer graph, the surfers can go to other objects by stay, in-link and out-link transitions, so all the possible paths are integrated. Figure 2 shows a random surfer graph derived from the citation network in Figure 1 with the retrieved similar objects to q_2 by RG-SimRank using $c = 0.8$, $\gamma = 0.5$ and $\lambda = 0.5$, in which the weights in the links correspond to the in-link transition probabilities. The neglected



Rank	RG-SimRank
1	(p_3, 0.333)
2	(p_1, 0.272)
3	(p_4,0.150)
4	(p_6, 0.137)
5	(p_5, 0.123)

Figure 2: Retrieved similar objects to query p_2 by RG-SimRank in random surfer graph

paths are now considered for finding similar objects. For example, the symmetric mixed-link path $p_2 \rightarrow p_1 \leftarrow p_5 \rightarrow p_4 \leftarrow p_3$ with "center" p_5 in the original network is considered by walking through the symmetric in-link path $p_2 \leftarrow p_1 \leftarrow p_5 \rightarrow p_4 \rightarrow p_3$ in the random surfer graph, which helps finding the neglected similar object p_3. There are also neglected dissymmetric out-link paths like $p_2 \rightarrow p_3 \leftarrow p_4 \leftarrow p_5$ are considered, which corresponds to symmetric in-link path $p_2 \rightarrow p_3 \leftarrow p_4 \leftarrow p_5 \curvearrowright p_5$ in the random surfer graph. Though the absolute scores of RG-SimRank and SimRank are different, the returned results mainly depend on the relative values and would not be affected.

Theorem 4.3. *(Comprehensiveness) When $0 < \gamma < 1$ and $0 < \lambda < 1$, RG-SimRank score $R_l(a,b)$ considers all the mixed-link paths between a and b of length $l_1 + l_2$ for $l_1, l_2 \le l$.*

Proof. When $a = b$, we have $R_0(a,b) = R_1(a,b) = ... = R_l(a,b) = 1$, so the theorem holds. When $a \ne b$, we assume $R_l(a,b)$ considers all the mixed-link paths $a = u_0 \dashrightarrow u_1 \dashrightarrow \ldots \dashrightarrow u_{l_1} = x = v_{l_2} \dashleftarrow v_{l_2-1} \dashleftarrow \ldots \dashleftarrow v_0 = b$ of length $l_1 + l_2$ for $l_1, l_2 \le l$. By Equation (7), we have

$$\begin{aligned} R_{l+1}(a,b) &= c \sum_{i=1}^{|\mathcal{I}(a)|} \sum_{j=1}^{|\mathcal{I}(b)|} \mathcal{P}_{a,\mathcal{I}_i(a)} \mathcal{P}_{b,\mathcal{I}_j(b)} R_l(\mathcal{I}_i(a), \mathcal{I}_j(b)) \\ &= c \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|\mathcal{I}(b)|} \mathcal{P}_{a,I_i(a)} \mathcal{P}_{b,\mathcal{I}_j(b)} R_l(I_i(a), \mathcal{I}_j(b)) \\ &\quad + c \sum_{j=1}^{|\mathcal{I}(b)|} \mathcal{P}_{a,a} \mathcal{P}_{b,\mathcal{I}_j(b)} R_l(a, \mathcal{I}_j(b)) \\ &= c \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} \mathcal{P}_{a,I_i(a)} \mathcal{P}_{b,I_j(b)} R_l(I_i(a), I_j(b)) \\ &\quad + c \sum_{i=1}^{|I(a)|} \mathcal{P}_{a,I_i(a)} \mathcal{P}_{b,b} R_l(I_i(a), b) \\ &\quad + c \sum_{j=1}^{|I(b)|} \mathcal{P}_{a,a} \mathcal{P}_{b,I_j(b)} R_l(a, I_j(b)) + c\mathcal{P}_{a,a} \mathcal{P}_{b,b} R_l(a,b) \end{aligned}$$

By Equation (4), we have $\mathcal{P}_{a,I_i(a)} = (1-\gamma)\lambda P'_{a,\mathcal{P}_{a,I_i(a)}} + (1-\gamma)(1-\lambda)P_{a,\mathcal{P}_{a,I_i(a)}}$, which contains the mixed-link transition $a \dashrightarrow I_i(a)$, and similarly $\mathcal{P}_{b,I_j(b)}$ contains the mixed-link transition $b \dashrightarrow I_j(b)$. By the assumption, we have

(1) $\sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} \mathcal{P}_{a,I_i(a)} \mathcal{P}_{b,I_j(b)} R_l(I_i(a), I_j(b))$ contains all the mixed-link paths $a \dashrightarrow u_0 = I_j(a) \dashrightarrow u_1 \dashrightarrow \ldots \dashrightarrow u_{l_1} = x = v_{l_2} \dashleftarrow v_{l_2-1} \dashleftarrow \ldots \dashleftarrow v_1 \dashleftarrow v_0 = I_j(b) \dashleftarrow b$, also denoted as $a = u'_0 \dashrightarrow u'_1 = I_j(a) \dashrightarrow u'_2 \dashrightarrow \ldots \dashrightarrow u'_{l'_1} = x = v'_{l'_2} \dashleftarrow v'_{l'_2-1} \dashleftarrow \ldots \dashleftarrow v'_2 \dashleftarrow v'_1 = I_j(b) \dashleftarrow v'_0 = b$ for $l'_1, l'_2 \le l + 1$;
(2) $\sum_{i=1}^{|I(a)|} \mathcal{P}_{a,I_i(a)} \mathcal{P}_{b,b} R_l(I_i(a), b)$ contains all the mixed-link paths $a \dashrightarrow u_0 = I_j(a) \dashrightarrow u_1 \dashrightarrow \ldots \dashrightarrow u_{l_1} = x = v_{l_2} \dashleftarrow v_{l_2-1} \dashleftarrow \ldots \dashleftarrow v_1 \dashleftarrow v_0 = b \curvearrowright b$, which is reduced to $a = u'_0 \dashrightarrow I_i(a) = u'_1 \dashrightarrow \ldots \dashrightarrow u'_{l'_1} = x = v'_{l'_2} \dashleftarrow v'_{l'_2-1} \dashleftarrow \ldots \dashleftarrow v'_1 \dashleftarrow v'_0 = b$ for $l'_1 \le l + 1$ and $l'_2 \le l$ by skipping $b \curvearrowright b$;
(3) $\sum_{j=1}^{|I(b)|} \mathcal{P}_{a,a} \mathcal{P}_{b,I_j(b)} R_l(a, I_j(b))$ contains all the mixed-link paths $a = u_0 \curvearrowleft a = u_1 \dashrightarrow u_2 \dashrightarrow \ldots \dashrightarrow u_{l_1} = x = v_{l_2} \dashleftarrow v_{l_2-1} \dashleftarrow \ldots \dashleftarrow v_2 \dashleftarrow v_1 = I_j(b) \dashleftarrow v_0 = b$, which is reduced to $a = u'_0 \dashrightarrow u'_1 \dashrightarrow \ldots \dashrightarrow u'_{l'_1} = x = v'_{l'_2} \dashleftarrow v'_{l'_2-1} \dashleftarrow \ldots v'_2 \dashleftarrow v'_1 = I_j(b) \dashleftarrow v'_0 = b$ for $l'_1 \le l$ and $l'_2 \le l + 1$ by skipping $a \curvearrowleft a$;
(4) $\mathcal{P}_{a,a} \mathcal{P}_{b,b} R_l(a,b)$ contains all the mixed-link paths $a = u_0 \curvearrowleft a = u_1 \dashrightarrow u_2 \dashrightarrow \ldots \dashrightarrow u_{l_1} = x = v_{l_2} \dashleftarrow v_{l_2-1} \dashleftarrow \ldots \dashleftarrow v_2 \dashleftarrow v_1 = b \curvearrowright v_0 = b$, which is reduced to $a = u'_0 \dashrightarrow u'_1 \dashrightarrow \ldots \dashrightarrow u'_{l'_1} = x = v'_{l'_2} \dashleftarrow v'_{l'_2-1} \dashleftarrow \ldots v'_1 \dashleftarrow v'_0 = b$ for $l'_1, l'_2 \le l$ similarly.

By replacing l'_1 and l'_2 with l_1 and l_2 respectively, we have $R_{l+1}(a,b)$ considers all the mixed-link paths of length $l_1 + l_2$ for $l_1, l_2 \le l + 1$. By induction, the theorem holds. □

4.2.3 *Inheritance of RG-SimRank.*

Since RG-SimRank is modeled as a same form of SimRank, and the random surfer graph that is also defined as a directed graph, so the mathematical properties of SimRank are obviously inherited, including the following important properties, described as:

Theorem 4.4. *For objects $a, b \in V$, stay probability $\gamma \in [0,1]$, backward probability $\lambda \in [0,1]$, decay factor $c \in (0,1)$ and path length l, we have*

(1) *(Symmetry) $R_l(a,b) = R_l(b,a)$.*
(2) *(Monotonicity) $0 \le R_l(a,b) \le R_{l+1}(a,b) \le 1$.*
(3) *(Existence) The solution to the iterative RG-SimRank equations exists and converges to a fixed point $R(*,*)$.*
(4) *(Uniqueness) The solution to the iterative RG-SimRank equations is unique when $c \ne 1$.*

This theorem can be proofed similar to SimRank [10] with discriminating the definition of transition probabilities. RG-SimRank is based on random surfer graph, in which the probabilities of stay, in- and out-link transitions are discriminated for similarity computation while SimRank treats the links equivalently. Other properties can be proofed similarly.

RG-SimRank inherits the properties of SimRank, so the optimization techniques on SimRank can be employed for speeding up the similarity computation. For example, the partial sums function [23] can be adopted by minor modification for transition probabilities, and the matrix form-based optimization techniques (*e.g.*, [17, 25]) can be adopted directly. Other type optimization techniques including single-source similarity computation (*e.g.*, [21, 31, 37]) and similarity join (*e.g.*, [50, 51]) are also applicable to our approach.

Table 1: Derivatives of RG-SimRank

$\gamma \backslash \lambda$	$\lambda = 0$	$0 < \lambda < 1$	$\lambda = 1$
$\gamma = 0$	rvs-SimRank	P-Rank, C-Rank	SimRank
$0 < \gamma < 1$	rvs-SimRank*	N/A	SimRank*
$\gamma = 1$	N/A	N/A	N/A

4.2.4 Derivatives of RG-SimRank. Besides the comprehensiveness and inheritance, RG-SimRank also outperforms other link-based similarity measures due to its generality. RG-SimRank enjoys the most general form of the structure perspective, under which lots of the similarity measures can be considered as its special cases, as shown in Table 1. When $\gamma = 0$, the stay transitions are not utilized, and we can set $\lambda = 1$ to consider only in-link transitions, which reduces RG-SimRank to SimRank; similarly, RG-SimRank is reduced to rvs-SimRank by setting $\lambda = 0$; and by setting $0 < \lambda < 1$, both in- and out-link transitions are considered, and then RG-SimRank is reduced to P-Rank and C-Rank. When $0 < \gamma < 1$, the stay transitions are considered, and we can further consider in-link directions by setting $\lambda = 1$, which reduces RG-SimRank to SimRank*; the out-link transitions can be considered by setting $\lambda = 0$, which reduces RG-SimRank to rvs-SimRank*; and for $0 < \lambda < 1$, there is no similarity measure of this kind. And when $\gamma = 1$, only stay transitions are considered, and RG-SimRank is reduced to a nonsense measure.

4.3 Enhancement of random surfer graph

Using random surfer graph, other link-based similarity measures besides SimRank can also be computed similarly in a comprehensive manner. The neglected similar objects would be returned by finding out the mixed-link paths of any length.

THEOREM 4.5. *When $0 < \gamma < 1$ and $0 < \lambda < 1$, a similarity measure, which considers path τ of length $l_1 + l_2$, would consider any path τ' between a and b of length $l'_1 + l'_2$ for $l'_1 \leq l_1 \wedge l'_2 \leq l_2$ by building a random surfer graph when computing the similarity between a and b.*

PROOF. Let τ be $u_0 \dashrightarrow u_1 \dashrightarrow \ldots \dashrightarrow u_{l_1} = x = v_{l_1} \dashleftarrow v_{l_1-1} \dashleftarrow \ldots \dashleftarrow v_0$, and τ' be $a = u'_0 \dashrightarrow u'_1 \dashrightarrow \ldots \dashrightarrow u'_{l'_1} = x' = v_{l'_1} \dashleftarrow v'_{l'_1-1} \dashleftarrow \ldots \dashleftarrow v'_0 = b$, where x and x' are the "center" objects. In the random surfer graph, we can extend paths of length l'_1 to l_1 and l'_2 to l_2 by staying at the nodes for $l_1 - l'_1$ and $l_2 - l'_2$ steps respectively; and for the link directions, according to the definition of random surfer graph, the link with any direction in τ can be mapped in τ'. Since τ is considered by the similarity metric, so τ' is utilized for computing similarity between a and b. □

By building a random surfer graph, more paths would be utilized for enhancing similarity computation of existing measures. For example, SimRank* [45] that considers only the in-link paths can be enhanced by utilizing the paths of different directions; P-Rank [49] and C-Rank [42] that consider only symmetric paths can be enhanced by utilizing the paths of any length; and RWR [36] can be enhanced by utilizing the paths of different lengths and directions. Similarly, the enhanced measures would inherit the mathematical properties of the original ones.

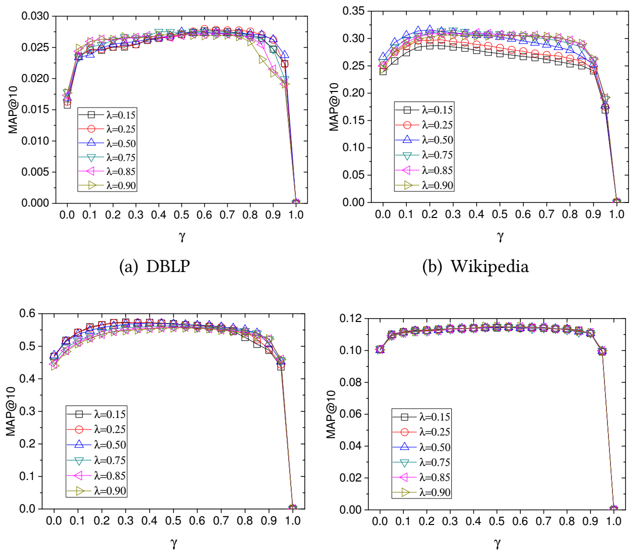
5 EXPERIMENTAL STUDY

5.1 Setup

5.1.1 Datasets. We use the following four datasets in the experiments. (1) DBLP. We extract a citation network from DBLP citation dataset [35], from which 10,686 papers and 13,097 citations are collected by breadth first search (BFS) over the links of "citation" relationship starting from a randomly chosen paper. (2) Wikipedia. We derive wikipedia network from Wikispeedia navigation path dataset [38] that consists of 4,604 articles and 119,864 hyper-links between these articles. (3) E-mail. The e-mail network is obtained from Email-Eu-core dataset [40] that is generated using e-mail data from a large European research institution. The e-mail network is composed of 1,005 users and 25,571 links of "delivery" relationship between these users. (4) Amazon. A co-purchased network is extracted from Amazon dataset [16]. We choose a product randomly and generate 10,927 products and 21,686 links of "co-purchased" relationship by BFS.

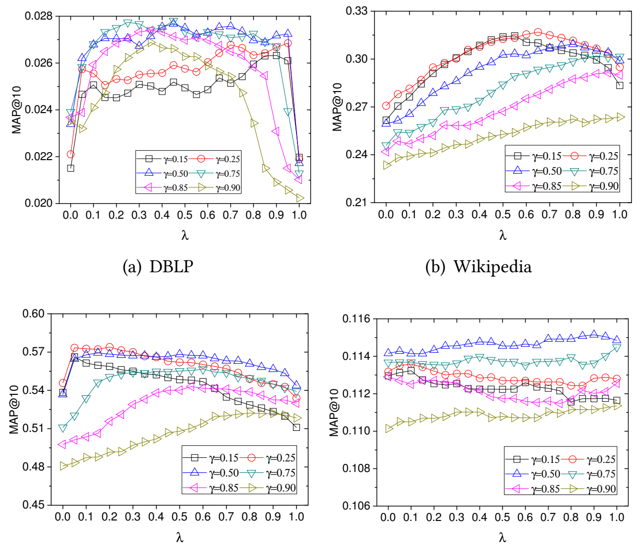
5.1.2 Comparison Methods and Evaluation. The following algorithms are implemented. (1) RG-SR, the proposed RG-SimRank; (2) SR, the SimRank algorithm [10]; (3) rvs-SR, the reverse SR; (4) SR*, the SimRank* algorithm [45]; (5) rvs-SR*, the reverse SR*; (6) PR, the P-Rank algorithm [49]; (7) CR, the C-Rank algorithm [42]; (8) RWR, the Random Walk with Restart algorithm [36]; and (9) rvs-RWR, the reverse RWR. SR, rvs-SR, PR and CR are speeded up by partial sums function [23], and SR* and rvs-SR* are speeded up by induced bipartite graph [45]. According to the literature, the decay factors of SR, rvs-SR, PR and CR are set as 0.8, the balance factor of PR is set as 0.5, and the restart probability of RWR is set as 0.8. Parameters γ and λ of RG-SR are set to be 0.5 if not specified explicitly. Our experiments are conducted on a 2.30 GHz Intel(R) Xeon(R) CPU with 12 GB RAM, running Windows 8.1, and all algorithms are implemented in C++ and compiled by using VS 2010.

The effectiveness is evaluated by the tasks of ranking and clustering respectively. We randomly pick 500 objects as queries from each dataset to retrieve similar objects, and the returned rankings are evaluated by mean average precision (MAP) that is defined as $\text{MAP}_{A,k} = \frac{1}{|Q|}\sum_{i=1}^{|Q|} \rho_{A,k}(v_i)$, where Q is a query set, A is a similarity measure, v_i is the i-th object in set Q, and $\rho_{A,k}(v_i)$ is the average precision of the returned top-k objects similar to v_i, which is defined as $\rho_{A,k}(v_i) = \frac{1}{k}\sum_{j=1}^{k} \varphi_A(v_i, v_j)$, where j is the rank of v_j, and $\varphi_A(v_i, v_j)$ is the precision function for evaluating the effectiveness of similarity between v_i and v_j. Precision function $\varphi_A(v_i, v_j)$ is defined similar to [10], in which the domain-specific properties that are not used for similarity computation are utilized as the ground truth. Specifically, in DBLP, $\varphi_{A,k}(v_i, v_j)$ is defined as the fraction of terms in v_j's title also in v_i's title; in Wikipedia, there are 129 hierarchical categories and each article might belong to different categories, by which $\varphi_{A,k}(v_i, v_j)$ is defined as the fraction of the categories contain article v_j also contain v_i; in E-mail, the community memberships are contained and each user belongs to exactly one of 42 departments, by which we define $\varphi_{A,k}(v_i, v_j)$ as 1 if users v_j and v_i are in a same department, and 0 otherwise; and in Amazon, there



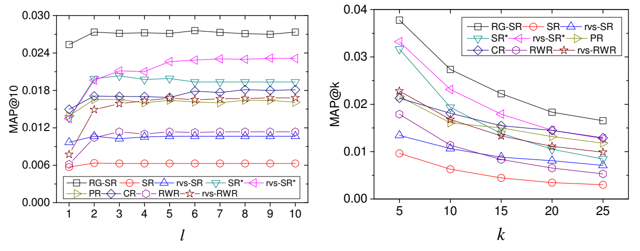
(a) DBLP (b) Wikipedia (c) E-mail (d) Amazon

Figure 3: MAP on varying γ



(a) DBLP (b) Wikipedia (c) E-mail (d) Amazon

Figure 4: MAP on varying λ



(a) On varying l (b) On varying k

Figure 5: MAP comparison in DBLP

are 9,711 hierarchical categories and each product might belong to different categories, by which $\varphi_{A,k}(v_i, v_j)$ is defined as the fraction of the categories contain product v_j also contain v_i.

We apply K-Medoids [13] to perform clustering based on the similarities returned by different algorithms. The clustering results are evaluated by compactness (CP) that is defined as $\text{CP}_{A,K} = \frac{(K-1)\sum_{i=1}^{K}\sum_{v_j \in C_i}\varphi(m_i,v_j)}{\sum_{i=1}^{K}\sum_{v_j \in C \setminus C_i}\varphi(m_i,v_j)}$, where K is the number of clusters to be generated, A is the similarity measure, $C = \cup_{i=1}^{K} C_i$, C_i is the i-th cluster, and m_i is the center of C_i. Intuitively, the numerator describes the precisions between centers and the objects that are grouped into the current clusters; and the denominator represents the precisions between centers and the objects that are in other clusters. By the definition of precision function, we know that a higher $\varphi(m_i, v_j)$ means m_i and v_j are more similar, so a higher $\text{CP}_{A,K}$ demonstrates a better clustering performance.

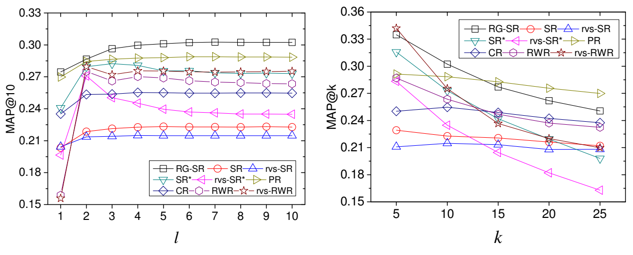
Efficiency comparison includes the running time of similarity computation and the sizes of non-zero similarity matrices.

5.2 Effectiveness

5.2.1 On Ranking Task. Figures 3 and 4 show the MAP scores on varying γ and λ respectively, where $l = 10$ and $k = 10$. In Figures 3, the curves for different datasets show a similar change as γ increases. At the beginning, the curves show an upward trend in a small range and become stable or change slowly in a long range, and then drops rapidly. In Figures 4, in DBLP, the MAP scores increase rapidly in a small range and then become slow as λ increases, and finally show a rapid downward. The MAP change of Wikipedia is not so evident as DBLP, this is might because there are some cyclic paths between articles while the paths between papers are acyclic, and similar change is shown in E-mail. In Amazon, the change trends are not evident though we set a smaller range in Y-axis for clear observation since the "co-purchased" relationship between products is bi-directional and the change of λ gives little affect to similarity computation. Empirically, when γ and λ are set as 0.5, the MAP scores are in a stable range.

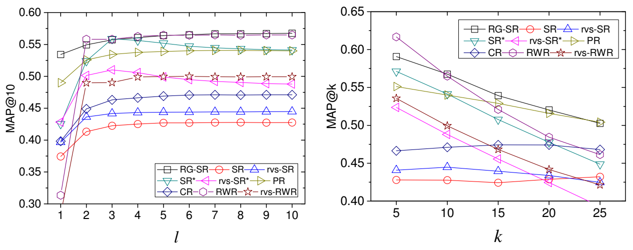
Figure 5(a) shows the MAP scores on varying l in DBLP, where $k = 10$. As l increases, the MAP scores of most algorithms increase and finally become stable, and the MAP score of RG-SR is always higher than other algorithms since it considers the mixed-link paths of any length. Generally, the MAP scores of the algorithms with considering more paths are relatively higher. For example, SR* shows a better performance than SR, since it considers the paths of any length, and similarly PR and CR are higher than SR and rsv-SR due to the consideration for the mixed-link paths.

Figure 5(b) shows the MAP scores on varying k in DBLP, where $l = 10$. We observe that the curves of all the algorithms show a downward trend as k increases. This is because the higher-ranking objects are more similar and should be closed to the given query, and the low-ranking objects should be relative in the rear order of the list of similar objects. The results demonstrate that RG-SR can not only find more objects similar to queries but also give a better quality of rankings.



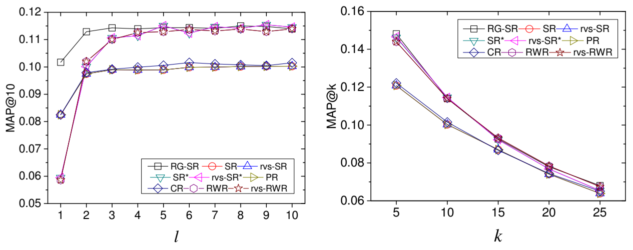
(a) On varying l (b) On varying k

Figure 6: MAP comparison in Wikipedia



(a) On varying l (b) On varying k

Figure 7: MAP comparison in E-mail



(a) On varying l (b) On varying k

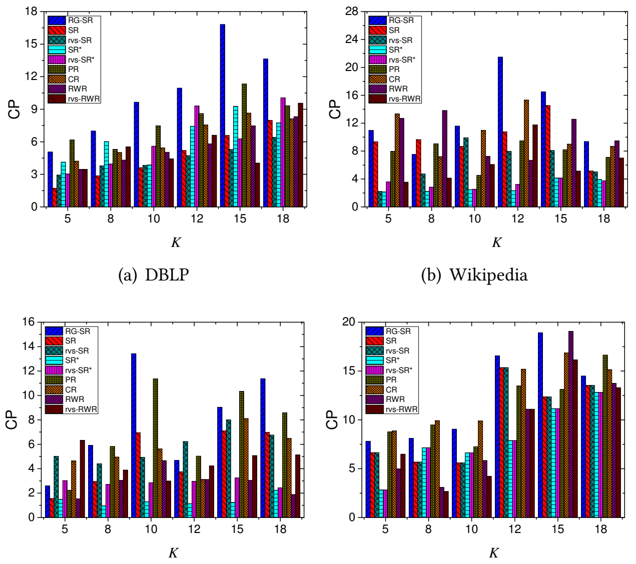
Figure 8: MAP comparison in Amazon

Figure 6(a) shows the MAP scores on varying l in Wikipedia, where $k = 10$. As l increases, RG-SR is always higher than other algorithms even the improvement is not so evident as DBLP. Differently, SR* has a downward trend from $l = 3$ since it considers only the paths of current iteration, and different iterations might produce different similarity scores, which does not guarantee monotonically increasing in theory, and rvs-SR* drops from $l = 3$ similarly.

Figure 6(b) shows the MAP scores on varying k in Wikipedia, where $l = 10$. Similar to DBLP, the curves of different algorithms show a downward trend as k increases. And the curve of RG-SR is higher than most of other algorithms. Generally, the superiority of RG-SR is evident except a minority of cases. For example, though the curve of RG-SR is lower than rvs-RWR when $k = 5$, it shows an evident superiority subsequently; and the curve of RG-SR is lower than PR when $k = 15, 20, 25$, but the MAP scores before $k = 10$ are evidently higher.

Figures 7(a) and 7(b) show the MAP change on varying l and k respectively in E-mail. In Figure 7(a), the curves of RG-SR and rvs-RWR are close as l increases. This is might because there are some mutual “delivery” relationship between users, which weakens the superiority of RG-SR. SR* and rvs-SR* drop slowly as l increases since the convergence is not guaranteed theoretically as mentioned. And the MAP change in Figure 7(b) is similar to Figure 6(b).

Figures 8(a) and 8(b) show the MAP change on varying l and k respectively in Amazon. In Figure 8(a), the curves of SR, rvs-SR, P-Rank, C-Rank, RWR and rvs-RWR are almost overlapped due to the bi-directional “co-purchased” relationship, and similarly SR* and rvs-SR* are overlapped. And SR* and rvs-SR* become close to RG-SR as l increases since they also consider the paths of different lengths. And in Figure 8(b), the MAP scores show a downward trend as k increases, which is consistent with the results of other datasets and can be explained similarly.



(a) DBLP (b) Wikipedia

(c) E-mail (d) Amazon

Figure 9: Compactness comparison

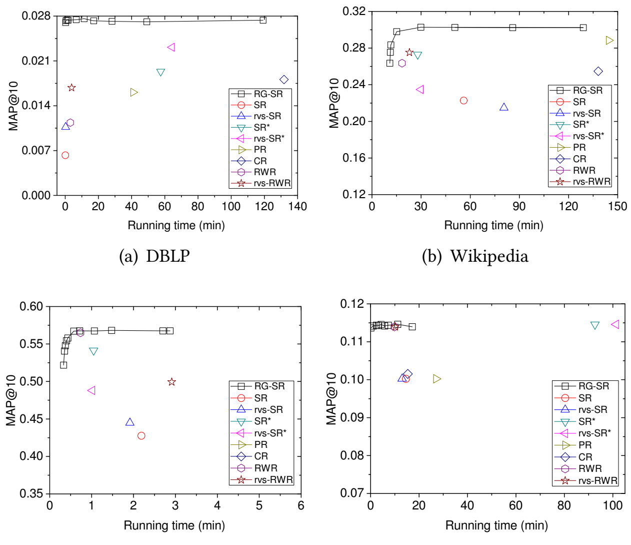
Table 2: MAP improvement on ranking task

Dataset	SR*	RG-SR*	PR	RG-PR	CR	RG-CR	RWR	RG-RWR
DBLP	0.019	**0.028**	0.016	**0.025**	0.018	**0.026**	0.011	**0.027**
Wikipedia	0.273	0.233	0.289	**0.302**	0.255	**0.273**	0.264	**0.289**
E-mail	0.546	**0.559**	0.563	**0.558**	0.485	**0.560**	0.565	**0.600**
Amazon	0.115	0.114	0.100	**0.113**	0.102	**0.114**	0.114	0.113

Table 3: CP improvement on ranking task

Dataset	SR*	RG-SR*	PR	RG-PR	CR	RG-CR	RWR	RG-RWR
DBLP	3.859	**4.811**	7.471	5.548	5.444	**9.703**	5.035	**7.957**
Wikipedia	2.466	2.204	4.514	**17.301**	10.984	**11.936**	7.258	**7.411**
E-mail	1.292	1.018	11.351	9.979	5.628	**8.871**	4.657	2.053
Amazon	6.626	**7.167**	7.244	**16.897**	9.885	8.157	5.828	**9.532**

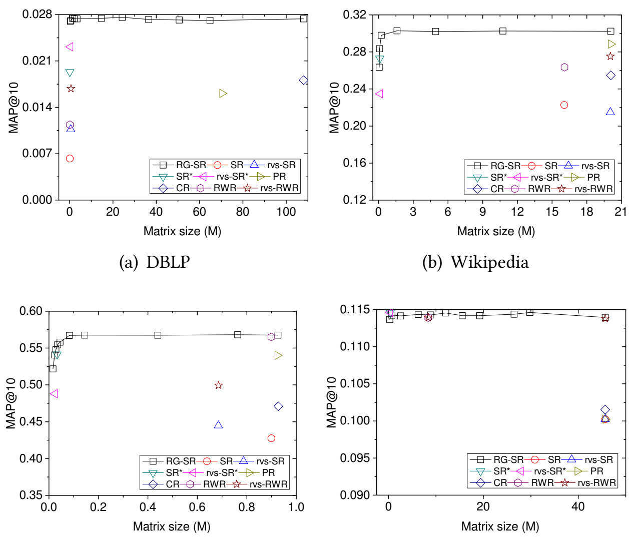
5.2.2 On Clustering Task. Figure 9(a), 9(b), 9(c) and 9(d) show the CP scores on different cluster number K. In most cases, the CP



(a) DBLP (b) Wikipedia

(c) E-mail (d) Amazon

Figure 10: MAP v.s. running time



(a) DBLP (b) Wikipedia

(c) E-mail (d) Amazon

Figure 11: MAP v.s. similarity matrix size

scores of RG-SR are evidently higher than the comparison algorithms. Though RG-SR is lower than a few of other algorithms in some cases, the differences are not so evident as the improvement. In Amazon, SR, SR* and RWR are close to their reverse forms at different K since the "co-purchased" relationship between products is bi-directional. Generally, RG-SR achieves better performances than the comparison algorithms, which suggests that the proposed approach not only do well in ranking objects but also can improve the effectiveness of clustering.

5.2.3 Using random surfer graph to enhance other metrics. We next use random surfer graph to enhance SR*, PR, CR and RWR, and the enhanced versions are denoted by RG-SR*, RG-PR, RG-CR and RG-RWR respectively. The MAP improvement on ranking task is shown in Table 2, where $l = 10$ and $k = 10$. In DBLP and E-mail, all the MAP scores are evidently improved; in Wikipedia, only the MAP score of SR* is decreased, this is might because the circles or circuits between articles disturb the similarity computation; and in Amazon, PR and CR are improved while SR* and RWR are decreased, since the bi-directional "co-purchased" relationship between products dilutes the enhancement of SR* and RWR, and both of them consider the paths of any length originally.

Table 3 shows CP improvement on clustering task, where $l = 10$ and $K = 10$. In DBLP, the CP scores of SR*, CR and RWR are evidently improved; in Wikipedia, the CP scores of PR, CR and RWR are improved; in E-mail, CR is enhanced; and in Amazon, only CR is decreased. Generally, in most cases, the quality of the results for ranking and clustering tasks is evidently improved by using random surfer graph.

5.3 Efficiency

Figures 10 and 11 show the MAP scores versus computation cost. During the similarity computation, we optimize RG-SR by the threshold-sieved approach in the literature [22], which speeds up the similarity computation by pruning similarity scores lower than a given threshold. Specifically, we tune the threshold from 0 to 0.05, and a higher threshold requires less computation cost. The leftmost ends of the curves of RG-SR correspond to the results when the threshold is set as 0, which is same to parameter settings in previous experimental setup. The parameter settings of the comparison algorithms are same to previous settings, and each algorithm corresponds to one point in the figures. When the threshold is set as 0, RG-SR requires higher time cost than most of the comparison algorithms since the mixed-link paths of any length are considered, but it still keeps a higher MAP score even the time costs are reduced. The results on the MAP scores versus matrix sizes can be analyzed similarly, in which symbol "M" in the caption refers to "1024*1024" and only the non-zero entries are stored.

6 CONCLUSION

This paper proposed RG-SimRank for comprehensively computing link-based similarities by building a random surfer graph. In contrast to existing similarity measures, RG-SimRank integrates all the possible paths of any direction and any length into similarity computation. With inheriting the mathematical properties, existing optimization techniques on SimRank are still applicable to the similarity computation. RG-SimRank provides a general solution to similarity computation, which makes lots of existing similarity measures become its special cases. And other similarity measures besides SimRank can also be enhanced similarly. Empirical studies on real datasets through comparison with the state-of-the-art similarity measures demonstrated the effectiveness and efficiency of the proposed approach.

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China under Grant 62002225, and Natural Science Foundation of Shanghai under Grant 21ZR1445400.

REFERENCES

[1] Leman Akoglu and Christos Faloutsos. 2009. RTG: a recursive realistic graph generator using random typing. *Data Min. Knowl. Discov.* 19, 2 (2009), 194–209.

[2] R. Amsler. 1972. *Application of citation-based automatic classification.* Technical report, The University of Texas at Austin Linguistics Research Center.

[3] Ioannis Antonellis, Hector Garcia-Molina, and Chi-Chao Chang. 2008. Simrank++: query rewriting through link analysis of the click graph. *PVLDB* 1, 1 (2008), 408–421.

[4] A. L. Barabasi and Reka Albert. 1999. Albert, R.: Emergence of Scaling in Random Networks. Science 286, 509-512. *Science* 286, 5439 (1999), 509–512.

[5] Yuanzhe Cai and Sharma Chakravarthy. 2011. Pairwise Similarity Calculation of Information Networks. In *DaWaK*. 316–329.

[6] Chih-Ming Chen, Chuan-Ju Wang, Ming-Feng Tsai, and Yi-Hsuan Yang. 2019. Collaborative Similarity Embedding for Recommender Systems. In *WWW*. 2637–2643.

[7] Mikhail Drobyshevskiy and Denis Turdakov. 2020. Random Graph Modeling: A Survey of the Concepts. *ACM Comput. Surv.* 52, 6 (2020), 131:1–131:36.

[8] Alan M. Frieze, Santosh S. Vempala, and Juan Vera. 2008. Logconcave random graphs. In *SOTC*. 779–788.

[9] Florian Geigl, Simon Walk, Markus Strohmaier, and Denis Helic. 2016. Steering the Random Surfer on Directed Webgraphs. In *WI*. 280–287.

[10] Glen Jeh and Jennifer Widom. 2002. SimRank: a measure of structural-context similarity. In *KDD*. 538–543.

[11] Glen Jeh and Jennifer Widom. 2003. Scaling personalized web search. In *WWW*. 271–279.

[12] Ruoming Jin, Victor E. Lee, and Hui Hong. 2011. Axiomatic ranking of network role similarity. In *KDD*. 922–930.

[13] Leonard Kaufman and Peter J. Rousseeuw. 1990. *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley.

[14] M. M. Kessler. 1963. Bibliographic coupling between scientific papers. *American Documentation* 14 (1963), 10–25.

[15] E. A. Leicht, Petter Holme, and M. E. J. Newman. 2006. Vertex similarity in networks. *Physical Review E* 73, 2 (2006), 026120–1–026120–10.

[16] Jure Leskovec, Lada A. Adamic, and Bernardo A. Huberman. 2007. The dynamics of viral marketing. *TWEB* 1, 1 (2007), 5.

[17] Cuiping Li, Jiawei Han, Guoming He, Xin Jin, Yizhou Sun, Yintao Yu, and Tianyi Wu. 2010. Fast computation of SimRank for static and dynamic information networks. In *EDBT*. 465–476.

[18] Zhenjiang Lin, Irwin King, and Michael R. Lyu. 2006. PageSim: A Novel Link-Based Similarity Measure for the World Wide Web. In *WI*. 687–693.

[19] Zhenjiang Lin, Michael R. Lyu, and Irwin King. 2007. Extending Link-based Algorithms for Similar Web Pages with Neighborhood Structure. In *WI*. 263–266.

[20] Zhenjiang Lin, Michael R. Lyu, and Irwin King. 2012. MatchSim: a novel similarity measure based on maximum neighborhood matching. *Knowl. Inf. Syst.* 32, 1 (2012), 141–166.

[21] Yu Liu, Bolong Zheng, Xiaodong He, Zhewei Wei, Xiaokui Xiao, Kai Zheng, and Jiaheng Lu. 2017. ProbeSim: Scalable Single-Source and Top-k SimRank Computations on Dynamic Graphs. *PVLDB* 11, 1 (2017), 14–26.

[22] Dmitry Lizorkin, Pavel Velikhov, Maxim N. Grinev, and Denis Turdakov. 2008. Accuracy estimate and optimization techniques for SimRank computation. *PVLDB* 1, 1 (2008), 422–433.

[23] Dmitry Lizorkin, Pavel Velikhov, Maxim N. Grinev, and Denis Turdakov. 2010. Accuracy estimate and optimization techniques for SimRank computation. *VLDB J.* 19, 1 (2010), 45–66.

[24] Tianshu Lyu, Yuan Zhang, and Yan Zhang. 2017. Enhancing the Network Embedding Quality with Structural Similarity. In *CIKM*. 147–156.

[25] Takanori Maehara, Mitsuru Kusumoto, and Ken-ichi Kawarabayashi. 2014. Efficient SimRank computation via linearization Publication of this article pending inquiry. In *KDD*. 1426–1435.

[26] Ana Gabriela Maguitman, Filippo Menczer, Fulya Erdinc, Heather Roinestad, and Alessandro Vespignani. 2006. Algorithmic Computation and Approximation of Semantic Similarity. *World Wide Web* 9, 4 (2006), 431–456.

[27] Shogo Murai and Yuichi Yoshida. 2019. Estimating Walk-Based Similarities Using Random Walk. In *WWW*. 1321–1331.

[28] Sai Kiran Narayanaswami, Balaraman Ravindran, and Venkatesh Ramaiyan. 2019. Generalized random Surfer-Pair models. In *ASONAM*. 452–455.

[29] Phuong Nguyen, Paolo Tomeo, Tommaso Di Noia, and Eugenio Di Sciascio. 2015. An evaluation of SimRank and Personalized PageRank to build a recommender system for the Web of Data. In *WWW*. 1477–1482.

[30] Chuan Shi, Xiangnan Kong, Yue Huang, Philip S. Yu, and Bin Wu. 2014. HeteSim: A General Framework for Relevance Measure in Heterogeneous Networks. *IEEE Trans. Knowl. Data Eng.* 26, 10 (2014), 2479–2492.

[31] Jieming Shi, Tianyuan Jin, Renchi Yang, Xiaokui Xiao, and Yin Yang. 2020. Real-time Index-Free Single Source SimRank Processing on Web-Scale Graphs. *Proc. VLDB Endow.* 13, 7 (2020), 966–978.

[32] Henry G. Small. 1973. Co-citation in the scientific literature: A new measure of the relationship betweenf two documents. *J. Am. Soc. Inf. Sci.* 24, 4 (1973), 265–269.

[33] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. 2011. PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks. *PVLDB* 4, 11 (2011), 992–1003.

[34] Marcin Sydow. 2004. Random surfer with back step. In *WWW*. 352–353.

[35] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. ArnetMiner: extraction and mining of academic social networks. In *KDD*. 990–998.

[36] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2008. Random walk with restart: fast solutions and applications. *Knowl. Inf. Syst.* 14, 3 (2008), 327–346.

[37] Zhewei Wei, Xiaodong He, Xiaokui Xiao, Sibo Wang, Yu Liu, Xiaoyong Du, and Ji-Rong Wen. 2019. PRSim: Sublinear Time SimRank Computation on Large Power-Law Graphs. In *SIGMOD*. 1042–1059.

[38] Robert West, Joelle Pineau, and Doina Precup. 2009. Wikispeedia: An Online Game for Inferring Semantic Distances between Concepts. In *IJCAI*. 1598–1603.

[39] Wensi Xi, Edward A. Fox, Weiguo Fan, Benyu Zhang, Zheng Chen, Jun Yan, and Dong Zhuang. 2005. SimFusion: measuring similarity using unified relationship matrix. In *SIGIR*. 130–137.

[40] Hao Yin, Austin R. Benson, Jure Leskovec, and David F. Gleich. 2017. Local Higher-Order Graph Clustering. In *KDD*. ACM, 555–564.

[41] Xiaoxin Yin, Jiawei Han, and Philip S. Yu. 2006. LinkClus: Efficient Clustering via Heterogeneous Semantic Links. In *PVLDB*. 427–438.

[42] Seok-Ho Yoon, Sang-Wook Kim, and Sunju Park. 2016. C-Rank: A link-based similarity measure for scientific literature databases. *Inf. Sci.* 326 (2016), 25–40.

[43] Weiren Yu, Xuemin Lin, and Wenjie Zhang. 2013. Towards efficient SimRank computation on large networks. In *ICDE*. 601–612.

[44] Weiren Yu, Xuemin Lin, and Wenjie Zhang. 2014. Fast incremental SimRank on link-evolving graphs. In *ICDE*. 304–315.

[45] Weiren Yu, Xuemin Lin, Wenjie Zhang, Lijun Chang, and Jian Pei. 2013. More is Simpler: Effectively and Efficiently Assessing Node-Pair Similarities Based on Hyperlinks. *PVLDB* 7, 1 (2013), 13–24.

[46] Mingxi Zhang, Hao Hu, Zhenying He, and Wei Wang. 2015. Top-k similarity search in heterogeneous information networks with x-star network schema. *Expert Syst. Appl.* 42, 2 (2015), 699–712.

[47] Mingxi Zhang, Jinhua Wang, and Wei Wang. 2018. HeteRank: A general similarity measure in heterogeneous information networks by integrating multi-type relationships. *Inf. Sci.* 453 (2018), 389–407.

[48] Zhipeng Zhang, Yingxia Shao, Bin Cui, and Ce Zhang. 2017. An Experimental Evaluation of SimRank-based Similarity Search Algorithms. *Proc. VLDB Endow.* 10, 5 (2017), 601–612.

[49] Peixiang Zhao, Jiawei Han, and Yizhou Sun. 2009. P-Rank: a comprehensive structural similarity measure over information networks. In *CIKM*. 553–562.

[50] Weiguo Zheng, Lei Zou, Lei Chen, and Dongyan Zhao. 2017. Efficient SimRank-Based Similarity Join. *ACM Trans. Database Syst.* 42, 3 (2017), 16:1–16:37.

[51] Weiguo Zheng, Lei Zou, Yansong Feng, Lei Chen, and Dongyan Zhao. 2013. Efficient SimRank-based Similarity Join Over Large Graphs. *PVLDB* 6, 7 (2013), 493–504.

[52] Yang Zhou, Hong Cheng, and Jeffrey Xu Yu. 2009. Graph Clustering Based on Structural/Attribute Similarities. *PVLDB* 2, 1 (2009), 718–729.