SimRank: A Measure of Structural-Context Similarity*

Glen Jeh
glenj@db.stanford.edu

Jennifer Widom
widom@db.stanford.edu

Stanford University

Abstract

The problem of measuring "similarity" of objects arises in many applications, and many domain-specific measures have been developed, e.g., matching text across documents or computing overlap among item-sets. We propose a complementary approach, applicable in any domain with object-to-object relationships, that measures similarity of the structural context in which objects occur, based on their relationships with other objects. Effectively, we compute a measure that says "two objects are similar if they are related to similar objects." This general similarity measure, called *SimRank*, is based on a simple and intuitive graph-theoretic model. For a given domain, SimRank can be combined with other domain-specific similarity measures. We suggest techniques for efficient computation of SimRank scores, and provide experimental results on two application domains showing the computational feasibility and effectiveness of our approach.

1 Introduction

Many applications require a measure of "similarity" between objects. One obvious example is the "find-similar-document" query, on traditional text corpora or the World-Wide Web [2]. More generally, a similarity measure can be used to cluster objects, such as for *collaborative filtering* in a recommender system [3, 6, 9], in which "similar" users and items are grouped based on the users' preferences.

Various aspects of objects can be used to determine similarity, usually depending on the domain and the appropriate definition of similarity for that domain. In a document corpus, matching text may be used, and for collaborative filtering, similar users may be identified by common preferences. We propose a general approach that exploits the object-to-object relationships found in many domains of interest. On the Web, for example, we can say that two pages are related if there are hyperlinks between them. A similar approach can be applied to scientific papers and their citations, or to any other document corpus with cross-reference information. In the case of recommender systems, a user's preference for an item constitutes a relationship between the user and the item. Such domains are naturally modeled as graphs, with nodes representing objects and edges representing relationships. We present an algorithm for analyzing the (logical) graphs derived from such data sets to compute similarity scores between nodes (objects) based on the *structural context* in which they appear, a concept to be made clear shortly. The intuition behind our algorithm is that, in many domains, *similar* objects are related to *similar* objects. More precisely, objects a and b are similar if they are related to objects c and d, respectively, and c and d are themselves similar. The base case is that objects are similar to themselves.

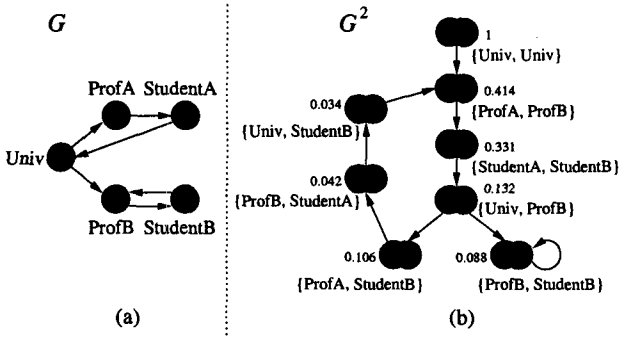

Figure 1: A small Web graph G and simplified node-pairs graph G^2. SimRank scores using parameter $C = 0.8$ are shown for nodes in G^2.

As an example, consider the tiny Web graph G shown in Figure 1(a), representing the Web pages of two professors ProfA and ProfB, their students StudentA and StudentB, and the home page of their university Univ. Edges between nodes represent hyperlinks from one page to another. From the fact that both are referenced (linked to) by Univ, we may infer that ProfA and ProfB are similar, and some previous algorithms are based on this *co-citation* [10] information. We generalize this idea by observing that once we have concluded similarity between ProfA and ProfB, and considering that ProfA and ProfB reference StudentA and StudentB respectively, we can also conclude that StudentA and StudentB are similar. Continuing forth, we can infer some similarity between Univ and ProfB, ProfA and StudentB, etc.

*This work was supported by the National Science Foundation under grants IIS-9817799 and IIS-9811947.

Let us logically represent the computation by using a node-pair graph G^2, in which each node represents an ordered pair of nodes of G. A node (a, b) of G^2 points to a node (c, d) if, in G, a points to c and b points to d. A simplified view of G^2 is shown in Figure 1(b); scores will be explained shortly. As we shall see later, scores are symmetric, so for clarity in the figure we draw (a, b) and (b, a) as a single node $\{a, b\}$ (with the union of their associated edges). Further simplifications in Figure 1(b) are explained in Section 3.

We run an iterative fixed-point algorithm on G^2 to compute what we call *SimRank* scores for the node-pairs in G^2. The SimRank score for a node v of G^2 gives a measure of similarity between the two nodes of G represented by v. Scores can be thought of as "flowing" from a node to its neighbors. Each iteration propagates scores one step forward along the direction of the edges, until the system stabilizes (i.e., scores converge). Since nodes of G^2 represent pairs in G, similarity is propagated from pair to pair. Under this computation, two objects are *similar* if they are referenced by *similar* objects.[1]

It is important to note that we are proposing a general algorithm that determines only the similarity of structural context. Our algorithm applies to any domain where there are enough relevant relationships between objects to base at least some notion of similarity on relationships. Obviously, similarity of other domain-specific aspects are important as well; these can—and should—be combined with relational structural-context similarity for an overall similarity measure. For example, for Web pages we can combine SimRank with traditional textual similarity; the same idea applies to scientific papers or other document corpora. For recommender systems, there may be built-in known similarities between items (e.g., both computers, both clothing, etc.), as well as similarities between users (e.g., same gender, same spending level). Again, these similarities can be combined with the similarity scores that we compute based on preference patterns, in order to produce an overall similarity measure.

The main contributions of this paper are as follows.

- A formal definition for *SimRank* similarity scoring over arbitrary graphs, several useful derivatives of SimRank, and an algorithm to compute SimRank scores (Section 3).
- A graph-theoretic model for SimRank that gives intuitive mathematical insight into its use and computation (Section 4).
- Experimental results using an initial in-memory implementation of SimRank over two different real data sets that show the effectiveness and feasibility of SimRank (Section 6 of the full version of this paper [4]).

Our basic graph model is presented in Section 2.

This paper is a shortened version of [4]. It primarily omits the discussion of related work and experimental results (Sections 2 and 6 of [4], respectively). It also omits some technical extensions and discussion, which are noted in the body of this paper.

2 Basic Graph Model

We model objects and relationships as a directed graph $G = (V, E)$ where nodes in V represent objects of the domain and edges in E represent relationships between objects. In Web pages or scientific papers, which are *homogeneous* domains, nodes represent documents, and a directed edge $\langle p, q \rangle$ from p to q corresponds to a reference (hyperlink or citation) from document p to document q. In a user-item domain, which is *bipartite*, we represent both users and items by nodes in V. A directed edge $\langle p, q \rangle$ corresponds to a purchase (or other expression of preference) of item q by person p. The result in this case is a bipartite graph, with users and items on either side. Note that edge weights may be used to represent varying degrees of preference, but currently they are not considered in our work.

For a node v in a graph, we denote by $I(v)$ and $O(v)$ the set of in-neighbors and out-neighbors of v, respectively. Individual in-neighbors are denoted as $I_i(v)$, for $1 \leq i \leq |I(v)|$, and individual out-neighbors are denoted as $O_i(v)$, for $1 \leq i \leq |O(v)|$.

3 SimRank

3.1 Motivation

Recall that the basic recursive intuition behind our approach is "two objects are *similar* if they are referenced by *similar* objects." As the base case, we consider an object maximally similar to itself, to which we can assign a similarity score of 1. (If other objects are known to be similar a-priori, such as from human input or text matching, their similarities can be preassigned as well.) Referring back to Figure 1, ProfA and ProfB are similar because they are both referenced by Univ (i.e., they are co-cited by Univ), and Univ is (maximally) similar to itself. Note in Figure 1(b) the similarity score of 1 on the node {Univ, Univ}, and the score of 0.414 on the node {ProfA, ProfB}. (How we obtained 0.414 will be described later.) StudentA and StudentB are similar because they are referenced by similar nodes ProfA and ProfB; notice the similarity score of 0.331 on the node for {StudentA, StudentB} in Figure 1(b).

In Section 3.2 we state and justify the basic equation that formalizes SimRank as motivated above. Section 3.3 modifies the equation for bipartite graphs, such as graphs for recommender systems as discussed in Section 2. The actual computation of SimRank values is discussed in Section 3.4, including pruning techniques to make the algorithm more efficient.

In Section 4.5 of the full version of this paper [4], we discuss the benefits of SimRank in scenarios where information is limited.

3.2 Basic SimRank Equation

Let us denote the similarity between objects a and b by $s(a, b) \in [0, 1]$. Following our earlier motivation, we write a recursive equation for $s(a, b)$. If $a = b$ then $s(a, b)$ is defined to be 1. Otherwise,

$$s(a, b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s(I_i(a), I_j(b)) \qquad (1)$$

where C is a constant between 0 and 1. A slight technicality here is that either a or b may not have any in-neighbors. Since we have no way to infer any similarity between a and b in this case, we should set $s(a, b) = 0$, so we define the summation in equation (1) to be 0 when $I(a) = \emptyset$ or $I(b) = \emptyset$.

One SimRank equation of the form (1) is written for each (ordered) pair of objects a and b, resulting in a set of n^2 SimRank

[1] The recursive nature of our algorithm, and thus its name, resembles that of the *PageRank* algorithm, used by the Google [1] Web search engine to compute importance scores for Web pages [8]. In [4] we discuss how PageRank and other iterative algorithms relate to our work.

equations for a graph of size n. Let us defer discussion of the constant C for now. Equation (1) says that to compute $s(a,b)$, we iterate over all in-neighbor pairs $(I_i(a), I_j(b))$ of (a,b), and sum up the similarity $s(I_i(a), I_j(b))$ of these pairs. Then we divide by the total number of in-neighbor pairs, $|I(a)||I(b)|$, to normalize. That is, the similarity between a and b is the average similarity between in-neighbors of a and in-neighbors of b. As discussed earlier, the similarity between an object and itself is defined to be 1.

It is shown in [4] that a simultaneous solution $s(*,*) \in [0,1]$ to the n^2 SimRank equations always exists and is unique. Thus we can define the *SimRank score* between two objects a and b to be the solution $s(a,b)$. From equation (1), it is easy to see that SimRank scores are symmetric, i.e., $s(a,b) = s(b,a)$.

We said in Section 1 that similarity can be thought of as "propagating" from pair to pair. To make this connection explicit, we consider the derived graph $G^2 = (V^2, E^2)$, where each node in $V^2 = V \times V$ represents a pair (a,b) of nodes in G, and an edge from (a,b) to (c,d) exists in E^2 iff the edges $\langle a,c\rangle$ and $\langle b,d\rangle$ exist in G. Figure 1(b) shows a simplified version of the derived graph G^2 for the graph G in Figure 1(a), along with similarity scores computed using $C = 0.8$. As mentioned earlier, we have drawn the symmetric pairs (a,b) and (b,a) as a single node $\{a,b\}$. Two types of nodes are omitted from the figure. The first are those *singleton* nodes which have no effect on the similarity of other nodes, such as {ProfA, ProfA}. The second are the nodes with 0 similarity, such as {ProfA, StudentA}.

Similarity propagates in G^2 from node to node (corresponding to propagation from pair to pair in G), with the sources of similarity being the singleton nodes. Notice that cycles in G^2, caused by the presence of cycles in G, allow similarity to flow in cycles, such as from {Univ, ProfB} back to {ProfA, ProfB} in the example. Similarity scores are thus *mutually reinforced.*

Now let us consider the constant C, which can be thought of either as a confidence level or a decay factor. Consider a simple scenario where page x references both c and d, so we conclude some similarity between c and d. The similarity of x with itself is 1, but we probably don't want to conclude that $s(c,d) = s(x,x) = 1$. Rather, we let $s(c,d) = C \cdot s(x,x)$, meaning that we are less confident about the similarity between c and d than we are between x and itself. The same argument holds when two distinct pages a and b cite c and d. Viewed in terms of similarity flowing in G^2, C gives the rate of decay (since $C < 1$) as similarity flows across edges. In Section 4 of this paper and in Section 6 of [4] we discuss the empirical significance of C.

Though we have given motivation for the basic SimRank equation, we have yet to characterize its solution, which we take to be a measure of similarity. It would be difficult to reason about similarity scores, to adjust parameters of the algorithm (so far only C), or to recognize the domains in which SimRank would be effective, if we cannot get an intuitive feel for the computed values. Section 4 addresses this issue with an intuitive model for SimRank.

We emphasize that the basic SimRank equation (and the bipartite version in 3.3) is but one way to encode mathematically our recursive notion of structural-context similarity. Another possibility is presented in Section 4.3.2 of the full version of this paper [4].

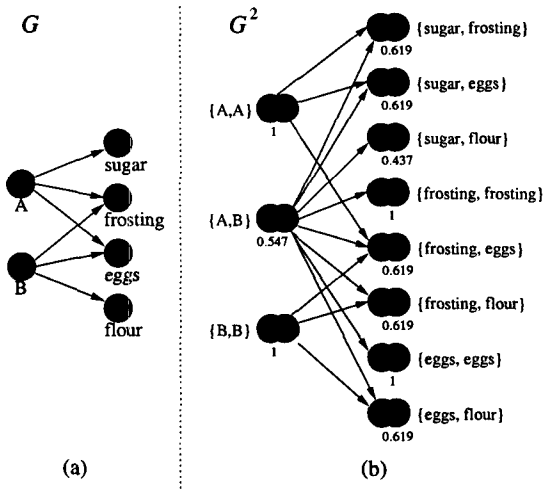

Figure 2: Shopping graph G and a simplified version of the derived node-pairs graph G^2. Bipartite SimRank scores are shown for G^2 using $C_1 = C_2 = 0.8$.

3.3 Bipartite SimRank

Next we extend the basic SimRank equation (1) to bipartite domains consisting of two types of objects. We continue to use recommender systems as motivation. Suppose persons A and B purchased itemsets {eggs, frosting, sugar} and {eggs, frosting, flour} respectively. A graph of these relationships is shown in Figure 2(a). Clearly, the two buyers are similar: both are baking a cake, say, and so a good recommendation to person A might be flour. One reason we can conclude that A and B are similar is that they both purchased eggs and frosting. But moreover, A purchased sugar while B purchased flour, and these are similar items, in the sense that they are purchased by similar people: cake-bakers like A and B. Here, similarity of items and similarity of people are mutually-reinforcing notions:

- People are *similar* if they purchase *similar* items.
- Items are *similar* if they are purchased by *similar* people.

The mutually-recursive equations that formalize these notions are analogous to equation (1). Let $s(A,B)$ denote the similarity between persons A and B, and let $s(c,d)$ denote the similarity between items c and d. Since, as discussed in Section 2, directed edges go from people to items, for $A \neq B$ we write the equation

$$s(A,B) = \frac{C_1}{|O(A)||O(B)|} \sum_{i=1}^{|O(A)|} \sum_{j=1}^{|O(B)|} s(O_i(A), O_j(B)) \quad (2)$$

and for $c \neq d$ we write

$$s(c,d) = \frac{C_2}{|I(c)||I(d)|} \sum_{i=1}^{|I(c)|} \sum_{j=1}^{|I(d)|} s(I_i(c), I_j(d)) \quad (3)$$

If $A = B$, $s(A,B) = 1$, and analogously for $s(c,d)$. Neglecting C_1 and C_2, equation (2) says that the similarity between persons A and B is the average similarity between the items they purchased, and equation (3) says that the similarity between items c and d is the average similarity between the people who purchased them. The constants C_1, C_2 have the same semantics as C in equation (1).

Figure 2(b) shows the derived node-pairs graph G^2 for the graph G in Figure 2(a). Simplifications have been made to G^2, as in Figure 1(b). Similarity scores for nodes of G^2, computed using $C_1 = C_2 = 0.8$, are also shown. Notice how sugar and flour are similar even though they were purchased by different people, although not as similar as, say, frosting and eggs. The node {frosting, eggs} has the same score as, say, {sugar, eggs}, even though frosting and eggs have been purchased together twice, versus once for sugar and eggs, since the normalization in equations (2) and (3) says that we consider only the percentage of times that items are purchased together, not the absolute number of times. It is, however, easy to incorporate the absolute number if desired; see Section 4.5 of [4].

3.3.1 Bipartite SimRank in Homogeneous Domains

It turns out that the bipartite SimRank equations (2) and (3) can also be applied to homogeneous domains, such as Web pages and scientific papers. Although a bipartite distinction is not explicit in these domains, it may be the case that elements take on different roles (e.g., "hub" pages and "authority" pages for importance [5]), or that in-references and out-references give different information. For example, two scientific papers might be similar as *survey* papers if they cite similar *result* papers, while two papers might be similar as result papers if they are cited by similar survey papers. In analogy with the HITS [5] algorithm, we can associate a "points-to" similarity score $s_1(a,b)$ to each pair of nodes a and b, as well as a "pointed-to" similarity score $s_2(a,b)$, and write the same equations (2) and (3) as if the domain were bipartite:

$$s_1(a,b) = \frac{C_1}{|O(a)||O(b)|} \sum_{i=1}^{|O(a)|} \sum_{j=1}^{|O(b)|} s_2(O_i(a), O_j(b))$$

$$s_2(a,b) = \frac{C_2}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s_1(I_i(a), I_j(b))$$

Depending on the domain and application, either score or a combination may be used.

3.4 Computing SimRank

3.4.1 Naive Method

A solution to the SimRank equations (or bipartite variations) for a graph G can be reached by iteration to a fixed-point. Let n be the number of nodes in G. For each iteration k, we can keep n^2 entries $R_k(*,*)$ of length n^2, where $R_k(a,b)$ gives the score between a and b on iteration k. We successively compute $R_{k+1}(*,*)$ based on $R_k(*,*)$. We start with $R_0(*,*)$ where each $R_0(a,b)$ is a lower bound on the actual SimRank score $s(a,b)$:

$$R_0(a,b) = \begin{cases} 0 & (\text{if } a \neq b) \\ 1 & (\text{if } a = b) \end{cases}$$

To compute $R_{k+1}(a,b)$ from $R_k(*,*)$, we use equation (1) to get:

$$R_{k+1}(a,b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} R_k(I_i(a), I_j(b)) \quad (4)$$

for $a \neq b$, and $R_{k+1}(a,b) = 1$ for $a = b$. That is, on each iteration $k+1$, we update the similarity of (a,b) using the similarity scores of the neighbors of (a,b) from the previous iteration k according to equation (1). The values $R_k(*,*)$ are nondecreasing as k increases. We show in [4] that they converge to limits satisfying (1), the SimRank scores $s(*,*)$, i.e., for all $a, b \in V$, $\lim_{k\to\infty} R_k(a,b) = s(a,b)$. In all of our experiments we have seen rapid convergence, with relative rankings stabilizing within 5 iterations (details are in Section [4]), so we may choose to fix a number $K \approx 5$ of iterations to perform.

Let us analyze the time and space requirements for this method of computing SimRank. The space required is simply $O(n^2)$ to store the results R_k. Let d_2 be the average of $|I(a)||I(b)|$ over all node-pairs (a,b). The time required is $O(Kn^2d_2)$, since on each iteration, the score of every node-pair (n^2 of these) is updated with values from its in-neighbor pairs (d_2 of these on average). As it corresponds roughly to the square of the average in-degree, d_2 is likely to be a constant with respect to n for many domains. The resource requirements for bipartite versions are similar.

We mentioned that typically $K \approx 5$, and in most cases we also expect the average in-degree to be relatively small. However, n^2 can be prohibitively large in some applications, such as the Web, where it exceeds the size of main memory. Specialized disk layout and indexing techniques may be needed in this case; such techniques are beyond the scope of this paper. However, in the next subsection we do briefly consider pruning techniques that reduce both the time and space requirements. Pruning has allowed us to run our experiments entirely in main memory, without the need for disk-based techniques.

3.4.2 Pruning

One way to reduce the resource requirements is to prune the logical graph G^2. So far we have assumed that all n^2 node-pairs of G^2 are considered, and a similarity score is computed for every node-pair. When n is significantly large, it is very likely that the neighborhood (say, nodes within a radius of 2 or 3) of a typical node will be a very small percentage ($< 1\%$) of the entire domain. Nodes far from a node v, whose neighborhood has little overlap with that of v, will tend to have lower similarity scores with v than nodes near v, an effect that will become intuitive in Section 4. Thus one pruning technique is to set the similarity between two nodes far apart to be 0, and consider node-pairs only for nodes which are near each other. If we consider only node-pairs within a radius of r from each other in the underlying undirected graph (other criteria are possible), and there are on average d_r such neighbors for a node, then there will be nd_r node-pairs. The time and space complexities become $O(Knd_rd_2)$ and $O(nd_r)$ respectively, where d_2 is the average of $|I(a)||I(b)|$ for pages a, b close enough to each other. Since d_r is likely to be much less than n and constant with respect to n for many types of data, we can think of the approximate algorithm as being linear with a possibly large constant factor.

Of course, the quality of the approximation needs to be verified experimentally for the actual data sets. For the case of scientific papers, our empirical results suggest that this is a good approximation strategy, and allows the computation to be carried out entirely in main memory for a corpus of $n = 278,626$ objects. More details can be found in Section [4].

4 Random Surfer-Pairs Model

As discussed in Section 3.2, it is important to have an intuition for the similarity scores produced by the algorithm. For this we provide

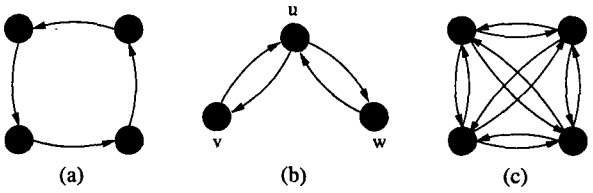

Figure 3: Sample graph structures.

an intuitive model based on "random surfers". We will show that the SimRank score $s(a,b)$ measures how soon two random surfers are expected to meet at the same node if they started at nodes a and b and randomly walked the graph backwards. The details involve some complexity, and are developed in the remainder of this section. The model is presented in the context of general directed graphs; variations for bipartite SimRank (Section 3.3) are easy to derive and we leave them to the interested reader.

4.1 Expected Distance

Let H be any strongly connected graph (in which a path exists between every two nodes). Let u, v be any two nodes in H. We define the *expected distance*[2] $d(u,v)$ from u to v as

$$d(u,v) = \sum_{t:u \rightsquigarrow v} P[t]l(t) \tag{5}$$

The summation is taken over all *tours* t (paths that may have cycles) which start at u and end at v, and do not touch v except at the end. For a tour $t = \langle w_1, \ldots, w_k \rangle$, the length $l(t)$ of t is $k-1$, the number of edges in t. The probability $P[t]$ of traveling t is $\prod_{i=1}^{k-1} \frac{1}{|O(w_i)|}$, or 1 if $l(t) = 0$. Note that the case where $u = v$, for which $d(u,v) = 0$, is a special case of (5): only one tour is in the summation, and it has length 0. Because of the presence of cycles, there are infinitely many tours from u to v, and (5) is an (convergent) infinite sum. The expected distance from u to v is exactly the expected number of steps a random surfer, who at each step follows a random out-edge, would take before he first reaches v, starting from u.

4.2 Expected Meeting Distance

For our model, we extend the concept of expected distance to *expected meeting distance* (*EMD*). Intuitively, the expected meeting distance $m(a,b)$ between a and b is the expected number of steps required before two surfers, one starting at a and the other at b, would meet if they walked (randomly) in lock-step. The EMD is symmetric by definition. Before formalizing EMD, let us consider a few examples. The EMD between any two distinct nodes in Figure 3(a) is (informally) ∞, since two surfers walking the loop in lock-step will follow each other forever. In Figure 3(b), $m(u,v) = m(u,w) = \infty$ (surfers will never meet) and $m(v,w) = 1$ (surfers meet on the next step), suggesting that v and w are much more similar to each other than u is to v or w. Between two distinct nodes of 3(c), the EMD is 3, suggesting a lower similarity than between v and w in 3(b), but higher than between u and v (or u and w).

[2]In the literature this quantity, in undirected graphs, is known as the *hitting time* [7], but we will develop the idea differently and so choose to use another name for our presentation.

To define EMD formally in G, we use the derived graph G^2 of node-pairs. Each node (a,b) of V^2 can be thought of as the present state of a pair of surfers in V, where an edge from (a,b) to (c,d) in G^2 says that in the original graph G, one surfer can move from a to c while the other moves from b to d. A tour in G^2 of length n represents a pair of tours in G also having length n.

The EMD $m(a,b)$ is simply the expected distance in G^2 from (a,b) to any singleton node $(x,x) \in V^2$, since singleton nodes in G^2 represent states where both surfers are at the same node. More precisely,

$$m(a,b) = \sum_{t:(a,b) \rightsquigarrow (x,x)} P[t]l(t) \tag{6}$$

The sum is taken over all tours t starting from (a,b) which touch a singleton node at the end and only at the end. Unfortunately, G^2 may not always be strongly connected (even if G is), and in such cases there may be no tours t for (a,b) in the summation (6). The intuitive definition for $m(a,b)$ in this case is ∞, as in Figure 3(b), discussed above. However, this definition would cause problems in defining distances for nodes from which some tours lead to singleton nodes while others lead to (a,b). We discuss a solution to this problem in the next section.

4.3 Expected-f Meeting Distance

There are various ways to circumvent the "infinite EMD" problem discussed in the previous section. For example, we can make each surfer "teleport" with a small probability to a random node in the graph (the solution suggested for PageRank in [8]). Our approach, which as we will see yields equations equivalent to the SimRank equations, is to map all distances to a finite interval: instead of computing expected length $l(t)$ of a tour, we can compute the expected $f(l(t))$, for a nonnegative, monotonic function f which is bounded on the domain $[0,\infty)$. With this replacement we get the *expected-f meeting distance*. For our purposes, we choose the exponential function $f(z) = c^z$, where $c \in (0,1)$ is a constant. The benefits of this choice of f, which has values in the range $(0,1]$ over domain $[0,\infty)$, are:

- Equations generated are simple and easy to solve.
- Closer nodes have a lower score (meeting distances of 0 go to 1 and distances of ∞ go to 0), matching our intuition of similarity.

We define $s'(a,b)$, the similarity between a and b in G based on expected-f meeting distance, as

$$s'(a,b) = \sum_{t:(a,b) \rightsquigarrow (x,x)} P[t]c^{l(t)} \tag{7}$$

where c is a constant in $(0,1)$. The summation is taken to be 0 if there is no tour from (a,b) to any singleton nodes. Note from (7) that $s'(a,b) \in [0,1]$ for all a, b, and that $s'(a,b) = 1$ if $a = b$.

Let us consider these similarity scores on Figure 3 using $C = 0.8$ as an example. Between any two distinct nodes a, b in Figure 3(a), $s'(a,b) = 0$. In Figure 3(b), $s'(v,w) = 0.8$ while $s'(u,v) = s'(u,w) = 0$. For any two distinct nodes in the complete graph of Figure 3(c), $s'(a,b) \approx 0.47$, a lower score than between v and w in Figure 3(b).

4.4 Equivalence to SimRank

We now show that $s'(*,*)$ exactly models our original definition of SimRank scores by showing that $s'(*,*)$ satisfies the SimRank

equations (1). To ease presentation, let us assume that all edges in our graph G have been reversed, so following an edge is equivalent to moving one step backwards in the original graph.[3]

First, to aid in understanding, we give an intuitive but informal argument about the expected distance $d(u,v)$ in a graph; the same ideas can be applied to the expected-f meeting distance. Suppose a surfer is at $u \in V$. At the next time step, he chooses one of $O_1(u), \ldots, O_{|O(u)|}(u)$, each with probability $\frac{1}{|O(u)|}$. Upon choosing $O_i(u)$, the expected number of steps he will still have to travel is $d(O_i(u), v)$ (the base case is when $O_i(u) = v$, for which $d(O_i(u), v) = 0$). Accounting for the step he travels to get to $O_i(u)$, we get:

$$d(u,v) = 1 + \frac{1}{|O(u)|} \sum_{i=1}^{|O(u)|} d(O_i(u), v)$$

With this intuition in mind, we derive similar recursive equations for $s'(a,b)$ which will show that $s'(a,b) = s(a,b)$. If $a = b$ then $s'(a,b) = s(a,b) = 1$. If there is no path in G^2 from (a,b) to any singleton nodes, in which case $s'(a,b) = 0$, it is easy to see from equation (4) that $s(a,b) = 0$ as well, since no similarity would flow to (a,b) (recall that edges have been reversed). Otherwise, consider the tours t from (a,b) to a singleton node in which the first step is to the out-neighbor $O_z((a,b))$. There is a one-to-one correspondence between such t and tours t' from $O_z((a,b))$ to a singleton node: for each t' we may derive a corresponding t by appending the edge $\langle (a,b), O_z((a,b)) \rangle$ at the beginning. Let T be the bijection that takes each t' to the corresponding t. If the length of t' is l, then the length of $t = T(t')$ is $l+1$. Moreover, the probability of traveling t is $P[t] = \frac{1}{|O((a,b))|} P[t'] = \frac{1}{|O(a)||O(b)|} P[t']$. We can now split the sum in (7) according to the first step of the tour t to write

$$\begin{aligned}
s'(a,b) &= \sum_{z=1}^{|O((a,b))|} \sum_{t' :\, O_z((a,b)) \rightsquigarrow (x,x)} P[T(t')] c^{l(T(t'))} \\
&= \sum_{z=1}^{|O((a,b))|} \sum_{t' :\, O_z((a,b)) \rightsquigarrow (x,x)} \frac{1}{|O(a)||O(b)|} P[t'] c^{l(t')+1} \\
&= \frac{c}{|O(a)||O(b)|} \sum_{z=1}^{|O((a,b))|} \sum_{t' :\, O_z((a,b)) \rightsquigarrow (x,x)} P[t'] c^{l(t)} \\
&= \frac{c}{|O(a)||O(b)|} \sum_{i=1}^{|O(a)|} \sum_{j=1}^{|O(b)|} s'(O_i(a), O_j(b)) \qquad (8)
\end{aligned}$$

Equation (8) is identical to the SimRank equation (1) with $c = C$ and in-edges swapped for out-edges. Since the solution to (1) is unique, $s'(a,b) = s(a,b)$ for all $a, b \in V$. Thus we have the following theorem.

Theorem. *The SimRank score, with parameter C, between two nodes is their expected-f meeting distance traveling back-edges, for $f(z) = C^z$.*

Thus, two nodes with a high SimRank score can be thought of as being "close" to a common "source" of similarity.

[3] Had we written equation (1) in terms of out-neighbors instead of in-neighbors, as may be appropriate in some domains, this step would not be necessary.

5 Future Work

There are a number of avenues for future work. Foremost, we must address efficiency and scalability issues, including additional pruning heuristics and disk-based algorithms. One possible approximation that differs from the neighborhood-based pruning heuristic in Section 3.4.2 is to divide a corpus into chunks, computing accurate similarity scores separately for each chunk and then combining them into a global solution. A second area of future work is to consider ternary (or more) relationships in computing structural-context similarity. For example, in the student-course domain we might also include the professors who taught the courses and the grades received by the students. Extending our entire framework to encompass such relationships should be possible, but it is not straightforward. Finally, we believe that structural-context similarity is only one component of similarity in most domains, so we plan to explore the combination of SimRank with other domain-specific similarity measures.

References

[1] `http://www.google.com`.

[2] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval.* Addison Wesley, Reading, Massachusetts, 1999.

[3] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, December 1992.

[4] Glen Jeh and Jennifer Widom. SimRank: A measure of structural-context similarity. Technical report, Stanford University Database Group, 2001. `http://dbpubs.stanford.edu/pub/2001-41`.

[5] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, California, January 1998.

[6] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. GroupLens: Applying collaborative filtering to Usenet news. *Communications of the ACM*, 40(3):77–87, March 1997.

[7] László Lovász. *Random Walks on Graphs: A Survey*, volume 2, pages 1–46. Bolyai Society Mathematical Studies, 1993.

[8] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the Web. Technical report, Stanford University Database Group, 1998. `http://citeseer.nj.nec.com/368196.html`.

[9] Upendra Shardanand and Pattie Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of the Conference on Human Factors in Computing Systems*, Denver, Colorado, 1995.

[10] Henry Small. Co-citation in the scientific literature: A new measure of the relationship between two documents. *Journal of the American Society for Information Science*, 24:265–269, 1973.