ASCOS: an Asymmetric Network Structure COntext Similarity Measure

Hung-Hsuan Chen[†], C. Lee Giles[†‡]
[†]Computer Science and Engineering, [‡]Information Sciences and Technology
The Pennsylvania State University, University Park, PA 16802, USA
hhchen@psu.edu, giles@ist.psu.edu

***Abstract*—Discovering similar objects in a social network has many interesting issues. Here, we present ASCOS, an Asymmetric Structure COntext Similarity measure that captures the similarity scores among any pairs of nodes in a network. The definition of ASCOS is similar to that of the well-known SimRank since both define score values recursively. However, we show that ASCOS outputs a more complete similarity score than SimRank because SimRank (and several of its variations, such as P-Rank and SimFusion) on average ignores half paths between nodes during calculation. To make ASCOS tractable in both computation time and memory usage, we propose two variations of ASCOS: a low rank approximation based approach and an iterative solver Gauss-Seidel for linear equations. When the target network is sparse, the run time and the required computing space of these variations are smaller than computing SimRank and ASCOS directly. In addition, the iterative solver divides the original network into several independent sub-systems so that a multi-core server or a distributed computing environment, such as MapReduce, can efficiently solve the problem. We compare the performance of ASCOS with other global structure based similarity measures, including SimRank, Katz, and LHN. The experimental results based on user evaluation suggest that ASCOS gives better results than other measures. In addition, the asymmetric property has the potential to identify the hierarchical structure of a network. Finally, variations of ASCOS (including one distributed variation) can also reduce computation both in space and time.**

I. Introduction

Complex network analysis has been a useful tool to describe the interaction and relations between pairs of objects. Studies of complex networks include observing the statistical properties of real networks [5], community detection [26], link discovery [4, 20, 21], etc. It has been applied on various research domains, including name disambiguation [24], biological networks analysis [8, 28], and personalized search [12].

To calculate the similarity score between a pair of nodes, one could use the attributes of nodes to infer their likeness. For example, consider a social network formed by students of an elementary school, two students are more likely to become friends if they have attributes in common such as same grade, same gender, interests, etc. This node attribute based method usually requires a large number of features to define a reasonable similarity score. However, collecting a good set of features sometimes requires domain experts. In addition, it could be difficult to define the similarity score when a few attributes are missing. We are interested in inferring the similarity between nodes based solely on structure context, i.e., the patterns of the edges. Structure context based similarity measures are attractive because in many cases the relationship between objects can be inferred without any domain knowledge. For example, genetic diseases caused by common genes can be inferred from the human disease network without using expensive biological experiments [8]. Future coauthoring behavior among scholars can be inferred by the coauthorship network without specifying the scholars' research interests or their IDs [7, 9, 20]. In addition, it has been found that structurally similar nodes on a social network are more likely to have similar node attributes and have similar behavior [27]. Some even suggest that structure based methods better explain user judgements than attribute based measures [23].

Among the structure context based similarity measures, SimRank [15] is probably the most influential and popular. Informally, SimRank defines the similarity score between two nodes i and j by similarity scores between i's neighbors and j's neighbors. This simple recursive definition makes it easy to implement and indirectly considers the structure of the entire network. Mathematically, SimRank is shown to be the same as measuring how soon two random surfers from i and j are expected to meet each other.

However, SimRank and its variations, such as P-Rank and SimFusion, have a problem which has surprisingly been neglected - they only measure the similarity between nodes that can reach each other in an even number of steps. Thus, for a large social network where two nodes in one connected component can usually reach each other by different paths, only the paths of even lengths contribute to the final SimRank score. The problem manifests itself in certain types of networks. Let's consider a simple network formed by two nodes (i and j) and one edge between them. Although we don't yet introduce the SimRank formula, one could easily understand that the SimRank score between i and j should be zero by the random surfer model: two random surfers starting at i and j would never meet each other because when one arrives i, the other must arrive j. Another example is the bi-partite graph formed by 2 sets of nodes V_1 and V_2 such that every edge in the graph joins a vertex in V_1 to a vertex in V_2. SimRank scores between neighbor nodes in the graph would always be zero, even though they should be related as neighbors. Again, one can easily see this with the random surfer analogy.

Several computational issues make the recursive definition based similarity measures intractable in practice. First, an ap-

plication may only need to know the similarity scores of only a few pairs of nodes. However, a recursive based definition needs to compute the scores between all pairs of nodes because the similarity between any pair of nodes depend on all other pairs of nodes. Second, storing the similarity scores for all pairs of nodes requires a large amount of memory. To compute the scores of a mid size network containing 0.1 million nodes, one would need $100,000 \times 100,000 \times 32$bit ≈ 40GB main memory to store the similarity scores (assuming the score between two nodes is floating point of 32 bits).

To deal with these issues, we propose ASCOS, an Asymmetric Structure COntext Similarity measure. Similar to SimRank, ASCOS defines similarity scores recursively so that the global network structure can be considered. Empirically, ASCOS is shown to return a better score than SimRank because ASCOS considers all paths between two target nodes, whereas SimRank considers only the paths of even lengths. To address the computational issues, ASCOS is reformulated into a non-recursive form. Two variations of ASCOS scoring and one distributed algorithm are proposed based on the non-recursive representation.

One interesting property of ASCOS is its asymmetric nature, i.e., similarity score from node i to node j may not equal the one from node j to node i. Traditionally, similarity measures are geometric: the target objects are projected into a multi-dimensional coordinate space, and the similarity score is proportional to the inverse of their geometric distance. Thus the similarity score behaves like a distance function which must be symmetric. However, Tversky discovered empirical evidence that systematically violates the symmetry assumption of similarity [31] and observed that the asymmetry appears to be determined, at least in part, by the relative salience of the objects to be compared. Users tend to select a more salient object as the referent and the less salient one as the subject. Here are a few examples that Tversky illustrated: *We say "the portrait resembles the person" rather than "the person resembles the portrait." We say "the son resembles the father" rather than "the father resembles the son."* Thus the judged similarity of portrait (son) to person (father) exceeds person (father) to portrait (son). We will later discuss more about the asymmetric nature and show how we introduce this salience property into ASCOS.

This paper makes the following contributions.

1) We show that SimRank and several of its variations suffer from the problem of considering only paths of even lengths. Therefore, on average half of the paths are excluded during the calculation.
2) We define a new similarity measure, ASCOS, which recursively considers the global structure of a network using all paths between nodes.
3) We propose two variations of ASCOS to address computational issues, namely time and memory limitations.
4) One of the two ASCOS variations can be applied on a distributed or multi-core environment, which is a significant advantage. This is a difficult for SimRank and the original ASCOS computation.
5) Among the popular similarity measures, ASCOS is one of the few that has an asymmetric property that can be useful for certain situations..
6) We conduct experiments on real networks using several global structure based similarity measures. User evaluation shows that ASCOS reports a better similarity score.

II. Related Works

Here we introduce local structure based similarity measures and global structure based similarity measures.

Local structure based similarity measures utilize local network structures to decide the similarity score between two nodes. Similarity scores computed by this type of measures are usually proportional to the number of mutual friends between two target nodes and can be written as $s_{ij} = (|N(i) \cap N(j)|)/C$, where $N(i)$ is the set of neighbors of node i, $|X|$ returns the number of elements of set X, and C is a normalizing constant whose value is determined by the specified similarity measure. For example, by Jaccard similarity the value of C is $|N(i) \cup N(j)|$ [30], by cosine similarity the value of C is $\sqrt{|N(i)||N(j)|}$ [29], by topology overlapping the value of C is $\min(|N(i)|, |N(j)|)$ [28]. The Adamic-Adar measure [2] intentionally assigns more weights to the vertices with fewer degrees, but it usually cannot be normalized. Another local structure based similarity measure, Preferential Attachment [5], defines the similarity score between nodes by multiplying the degrees of two nodes. Empirical studies show that Preferential Attachment usually reports a poor performance. Comprehensive studies of local structure based measures can be found in [11, 35].

Global structure based measures consider the structure of the whole network to determine the similarity scores between pairs of nodes. For example, the Katz similarity is based on the total number of paths between two nodes where longer paths are assigned lower weights [16]. Compared to two low degree nodes, two nodes with very high degrees are more likely to have one or several paths of a fixed length ℓ between them. As a result, high degree nodes tend to be more similar to every other node by Katz measure. To address the problem, LHN [18] suggested normalizing the number of paths of length ℓ by the expected number of such paths given the degrees of nodes. SimRank [15] is probably the most famous global structure based similarity measure. The similarity score between two nodes i and j depends on similarity scores between i's neighbors and j's neighbors, as defined in Equation 1.

$$s_{ij} := \begin{cases} \frac{c}{|N_{in}(i)||Nin(j)|} \sum_{\forall k \in N_{in}(i)} \sum_{\forall \ell \in N_{in}(j)} s_{k\ell} & \text{if } i \neq j \\ 1 & \text{if } i = j, \end{cases} \quad (1)$$

where c is the discounted parameter to control the relative importance between neighbors and in-direct neighbors, and $N_{in}(i)$ is the set of in-neighbors of node i. Several methods are influenced by SimRank. For example, P-Rank [34] extends SimRank by considering both in-neighbors and out-neighbors. SimFusion [32] supports different intra-node relations and

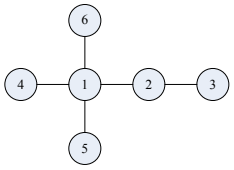


Fig. 1. The toy network.

different edge weights. The relationship between SimRank, P-Rank, and SimFusion is discussed in [6].

Global structure based similarity measure usually requires a great deal of computation. Several methods were proposed to approximate these measures. For the Katz score, a truncated spectral decomposition method was proposed [1]. For SimRank, an approximation measure [19] and a parallel computation [14] were investigated. Yu et al. proposed SimRank+ [33] to prevent a divergence issue and faster computation. A comprehensive survey on both local structure based and global structure based similarity measures can be found in [22].

III. ASCOS Framework

A. ASCOS Score Introduction

The SimRank measure states that two nodes i and j are similar if the in-neighbors of i and the in-neighbors of j are themselves similar. The recursive definition employs the entire network topology in calculating the similarity scores. However, such a definition fails to capture the relationship between nodes that can only reach others in an odd number of steps, as pointed out in Section I. In an extreme case, two neighboring nodes can have zero similarity score. This is counter-intuitive because neighboring nodes should have something in common if directly connected.

Instead of defining the similarity score between nodes by the relationship between both of the nodes' in-neighbors, ASCOS states that the similarity score from a node i to a node j is dependent on the similarity score from node i's in-neighbors to node j. This statement has two interesting properties. First, the out-neighbors are not involved. Second, the similarity score is asymmetric because it considers the in-neighbors of i but not the in-neighbors of j. We justify these settings below.

We consider only the in-neighbors because an object is not defined by how it describes others but by how others describe it. This idea is very similar to PageRank since the importance of a page is determined by its incoming pages not by the ones it points to. However, this definition can be easily extended to consider both in-neighbors and out-neighbors.

Traditionally, a similarity measure defines the similarity function to be proportional to the inverse of the distance function, which is usually calculated by projecting the target objects into a n-dimensional coordinate system and measuring their distance. Thus, the similarity function should be symmetric, i.e., $s_{ij} = s_{ji}$. By the definition, a statement of "*a* is like *b*" and a statement of "*b* is like a" should be of equal value. However, studies have shown that dimensional representations are not appropriate for some objects, like faces, countries, or personalities [31]. People tend to be more positive to "*a* is like *b*" than "*b* is like *a*" when *b* is more salient or general than *a* [31]. For example, "an ellipse is like a circle" is more likely to be true psychologically than "a circle is like an ellipse".

Another way to look at the similarity of nodes in a network is to measure the tendency to form a link between nodes. As suggested in [7], when modeling the coauthoring behavior as a coauthorship network, a young researcher is usually more eager to establish connections with strong researchers than the other way around. The similarity score, which is a proxy to measure the tendency of link formation, is apparently asymmetric because a young researcher is more willing to establish a link to an experienced researcher than vise versa.

To make the asymmetry concept clearer, let's examine node 1 and node 4 of Figure 1. Node 4 has only one neighbor node 1, but node 1 has four neighbors node 2, node 4, node 5, and node 6. In such a scenario people tend to be more positive to "node 4 is similar to node 1" than "node 1 is similar to node 4" because node 4's only neighbor is node 1 but node 1 has three other alternative options.

Algorithm 1: Naïve ASCOS calculation

Input: $\mathbf{A}$: an adjacency matrix of size n by n; c: the discounted parameter
Output: $\mathbf{S} = [s_{ij}]$: ASCOS similarity score matrix

```
S ← initial guessing matrix of size n by n;
while S not converge do
    for i ← 1 to n do
        for j ← 1 to n do
            Update s_ij by Equation 2;
        end
    end
end
```

B. Naïve ASCOS Calculation

We define the similarity value from i to j to be the discounted cumulative similarity score from all i's neighbors to j. ASCOS score s_{ij} from i to j can be written as follows.

$$s_{ij} := \begin{cases} \frac{c}{|N_{in}(i)|} \sum_{\forall k \in N_{in}(i)} s_{kj} & \text{if } i \neq j \\ 1 & \text{if } i = j, \end{cases} \quad (2)$$

where $N_{in}(i)$ is the set of in-neighbors of node i.

The relative importance parameter c is between 0 and 1. It controls the relative importance between the direct neighbors and indirect neighbors, i.e., neighbors' neighbors. The smaller the value, the less important the indirect neighbors are.

Algorithm 1 lists the pseudo code of ASCOS calculation based on the recursive definition.

C. Analysis of Naïve ASCOS

Let $\bar{N}_{in}$ denotes the average in-degree of the target network and k represents the required iteration for $\mathbf{S} = [s_{ij}]$ to converge, Algorithm 1 needs $O(k\bar{N}_{in}n^2) \approx O(n^2)$ computation time (assuming $k \ll n, \bar{N}_{in} \ll n$). Although the square computation time is infeasible in real-time, in practice this is a minor problem because the scores can be pre-computed offline. The more serious problem is the space complexity. The

recursive definition requires the computer to store all entries of $\mathbf{S}$ during computation thus $O(n^2)$ space is needed. For modern social networks which usually have tens to hundreds of millions of nodes, allocating the space that is proportional to the square of the number of nodes can be intractable.

IV. Efficient ASCOS Calculation

Naïve ASCOS calculation is inefficient in both time and space. To conquer these problems, two efficient variations, singular value decomposition based low rank approximation and recursive solver for systems of linear algebraic equations, are proposed. The later one can further be designed for a multi-core or distributed environment. As a result, applying ASCOS on a large scale social network is possible.

The root reason of keeping the entire $\mathbf{S}$ matrix is because the recursive definition of ASCOS that makes each s_{ij} be dependent on every other similarity score. To break the dependency, we redefine ASCOS as a non-recursive score and then propose two algorithms to calculate it.

A. Non-recursive ASCOS Equation

Given a graph G and its adjacency matrix $\mathbf{A} = [a_{ij}]$, we calculate $\mathbf{P} = [p_{ij}]$ as the column-normalized matrix of $\mathbf{A}$, i.e., $p_{ij} = a_{ij} / \sum_{\forall k} a_{kj}$. When we iterate Equation 2 to a sufficiently large number of times, it can be re-written as a matrix form, as shown in Equation 3.

$$\mathbf{S} = c\mathbf{P}^T\mathbf{S} + (1-c)\mathbf{I}, \tag{3}$$

where $\mathbf{P}^T$ is the transpose of $\mathbf{P}$.

The solution for Equation 3 can be computed by

$$\mathbf{S} = (1-c)(\mathbf{I} - c\mathbf{P}^T)^{-1}. \tag{4}$$

Now we turn ASCOS into a non-recursive equation. Note there is a subtle difference between Equation 2 and Equation 4 because the diagonal entries of $\mathbf{S}$ in Equation 4 is not set to one. However, this problem is trivial because it only effects the absolute similarity scores but not the relative relationship between similarity scores, i.e., if by Equation 2 we get $s_{ij} > s_{ik}$, the relation still holds when using Equation 4 instead.

Although Equation 4 is in a non-recursive form and therefore breaks the dependency limitation, the space complexity is still $O(n^2)$ and the time complexity even increases to an intractable $O(n^3)$ because of the matrix inverse operation. We now propose two methods to solve these problems.

B. Low Rank Approximation

Let $\mathbf{Q} = \mathbf{I} - c\mathbf{P}^T$. Although Equation 4 avoids recursive calculation, solving the inverse of matrix $\mathbf{Q}$ is challenging in general. First, $\mathbf{Q}$ could be a singular matrix, i.e., it is non-invertible. Second, calculating the inverse of a matrix requires a cubic computation time. Third, although $\mathbf{Q}$ is usually a sparse matrix, $\mathbf{Q}^{-1}$ is very likely to be a dense matrix. As we discussed earlier it is impracticable to fit the entire n by n matrix into the memory. In this section, we introduce low-rank approximation to avoid calculating and storing $\mathbf{Q}^{-1}$ directly.

Algorithm 2: ASCOS calculation by low rank approximation

Input: $\mathbf{A}$: an adjacency matrix of size n by n; c: the discounted parameter
Output: $\mathbf{S} = [s_{ij}]$: ASCOS similarity score matrix

```
P ← ColumnNormalize(A);
Q ← I − cP^T;
Do SVD for Q to obtain Ũ, Σ̃, and Ṽ^T;
Calculate S by Equation 5;
```

By Singular Value Decomposition (SVD), the matrix $\mathbf{Q}$ is factorized into $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices and $\mathbf{\Sigma}$ is a diagonal matrix. An approximation matrix $\tilde{\mathbf{Q}}$ of $\mathbf{Q}$ can be derived by $\tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^T$, in which $\tilde{\mathbf{\Sigma}}$ is a diagonal matrix by keeping the largest r singular values in $\mathbf{\Sigma}$, $\tilde{\mathbf{U}}$ is the first r columns of $\mathbf{U}$, and $\tilde{\mathbf{V}}^T$ is the first r rows of $\mathbf{V}^T$. Among all the matrices with rank r, $\tilde{\mathbf{Q}}$ is the one with minimum Frobenius norm difference to $\mathbf{Q}$. The value of $\mathbf{Q}^{-1}$ is approximated by $\tilde{\mathbf{Q}}^{-1} = \tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{U}}^T$. In practice we use Lanczos Algorithm [10] to perform SVD for such a large sparse matrix $\mathbf{Q}$. We can re-write Equation 4 to get $\mathbf{S}$.

$$\mathbf{S} \approx (1-c)\tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{U}}^T. \tag{5}$$

Since $\tilde{\mathbf{\Sigma}}$ is a diagonal matrix, $\tilde{\mathbf{\Sigma}}^{-1}$ can be easily calculated by inverting the diagonal entries from σ_{ii} to $1/\sigma_{ii}$.

Low rank approximation solves most of the problems we may face when calculating $\mathbf{Q}^{-1}$. First, since every matrix can be factorized by SVD, we don't need to worry about the singular matrix problem. Second, low rank approximation is space efficient. Assuming $r << n$, the matrix $\tilde{\mathbf{V}}^T$ and $\tilde{\mathbf{U}}$ each needs $O(rn)$ space. The matrix $\tilde{\mathbf{\Sigma}}^{-1}$ is a diagonal matrix with rank r that needs only $O(r)$ space. Thus overall only $O(n)$ space complexity is required. In addition, the time complexity is improved from cubic to square [13].

One assumption hidden behind the low rank approximation is the existence of notable linear correlations in $\mathbf{Q}$. The assumption seems correct because real world networks tend to have a high clustering coefficient, i.e., similar nodes tend to connect to the same set of nodes. However, this assumption cannot be guaranteed. Once the linear correlations are not manifest in the matrix, a low rank approximation could be less accurate. In addition, it is not straightforward to decide the value of the parameter r.

C. Gauss-Seidel (GS) Approach

While low rank approximation approximates $\mathbf{Q}^{-1}$ efficiently, obtaining $\mathbf{Q}^{-1}$ is not our final goal. We want $\mathbf{Q}^{-1}$ because it helps get $\mathbf{S}$. In this section, we introduce the approach to get $\mathbf{S}$ without calculating $\mathbf{Q}^{-1}$ and without the assumption of existence of linear correlation in $\mathbf{Q}$.

Let's split $\mathbf{S}$ into n column vectors $\mathbf{S} = [\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_n]$ and the identity matrix $\mathbf{I}$ into n column vectors $\mathbf{I} = [\mathbf{I}_1, \mathbf{I}_2, ..., \mathbf{I}_n]$. Equation 3 can be re-written into the following form.

$$(\mathbf{I} - c\mathbf{P}^T)\mathbf{S}_i = (1-c)\mathbf{I}_i, \tag{6}$$

Algorithm 3: ASCOS calculation by Gauss-Seidel

Input: $\mathbf{A}$: an adjacency matrix of size n by n; c: the discounted parameter
Output: $\mathbf{S}_i = [s_1, s_2, ..., s_n]^T$: ASCOS similarity score of node i

$\mathbf{P} \leftarrow$ ColumnNormalize($\mathbf{A}$);
$\mathbf{Q} \leftarrow \mathbf{I} - c\mathbf{P}^T$;
$\mathbf{S}_i \leftarrow$ initial guessing vector of size n;
$\mathbf{B}_i \leftarrow (1-c)I_i$;
while $\mathbf{S}_i$ *not converge* **do**
  for $j \leftarrow 1$ **to** n **do**
    Update s_j by Equation 8;
  end
end

where $i = 1, 2, ..., n$.

This turns the problem into a classic systems of linear algebraic equations, in which $\mathbf{I} - c\mathbf{P}^T$, or briefly $\mathbf{Q}$, is a coefficient matrix of dimension n by n; $(1-c)\mathbf{I}_i$, or briefly $\mathbf{B}_i$, is a constant column vector of size n; $\mathbf{S}_i$ is an unknown column vector. Such transformation allows us to compute the similarity score between two nodes by only calculating $1/n$ of the adjacency matrix instead of the entire matrix.

To solve the systems of linear algebraic equations, a standard tool is Gaussian elimination. However, Gauss elimination becomes inefficient and sometimes inapplicable when the coefficient matrix is sparse and the number of unknowns is large. Instead, we apply the Gauss-Seidel (GS) method, a recursive algorithm that repeatedly improves the solution until the unknown $\mathbf{S}_i$ converges.

Let s_j be the j^{th} element of $\mathbf{S}_i$, q_{jk} be the $(j,k)^{th}$ entry of matrix $\mathbf{Q}$, and b_j be the j^{th} element of $\mathbf{B}_i$. Equation 6 in scalar notation is written as

$$\sum_{k=1}^{n} q_{jk} s_k = b_j, \tag{7}$$

where $j = 1, 2, ..., n$.

The value of s_j in Equation 7 is solved by Equation 8.

$$s_j = \frac{1}{q_{jj}} \left(b_j - \sum_{\forall k \neq j} q_{jk} s_k \right). \tag{8}$$

The GS algorithm starts by randomly initializing the vector $\mathbf{S}_i$, and then iteratively updates each element of $\mathbf{S}_i$ by Equation 8 until $\mathbf{S}_i$ converges, as shown in Algorithm 3.

Note that for GS algorithm, the convergence of $\mathbf{S}_i$ is guaranteed only if either the coefficient matrix $\mathbf{Q}$ is symmetric positive-definite or $\mathbf{Q}$ is diagonally dominant, i.e., the magnitude of the diagonal entry in a row is no less than the sum of the magnitude of the non-diagonal entries. To show that $\mathbf{Q}$ is diagonally dominant, we re-write $\mathbf{Q}$ in the scalar notation.

$$q_{ij} = \begin{cases} 1 & \text{if } i = j \\ -cp_{ji} & \text{if } i \neq j. \end{cases} \tag{9}$$

Algorithm 4: Mapper of ASCOS Gauss-Seidel

Input: $\mathbf{A}$: an adjacency matrix of size n by n; c: the discounted parameter

$\mathbf{P} \leftarrow$ ColumnNormalize($\mathbf{A}$);
$\mathbf{Q} \leftarrow \mathbf{I} - c\mathbf{P}^T$;
for $i \leftarrow 1$ **to** n **do**
  $\mathbf{S}_i \leftarrow$ initial guessing vector of size n;
  $\mathbf{B}_i \leftarrow (1-c)I_i$;
  Emit(i, $\mathbf{Q}$, $\mathbf{S}_i$, $\mathbf{B}_i$) ; `// i: key to Reducer`
end

Algorithm 5: Reducer of ASCOS Gauss-Seidel

while $\mathbf{S}_i$ *not converge* **do**
  for $j \leftarrow 1$ **to** n **do**
    Update s_j by Equation 8;
  end
end
Emit($\mathbf{S}_i$);

The matrix $\mathbf{Q}$ is a diagonal dominant matrix because

$$\begin{aligned} \textstyle\sum_{\forall j \neq i} |q_{ij}| &= \textstyle\sum_{\forall j \neq i} |-cp_{ji}| \\ &\leq \textstyle\sum_{\forall j \neq i} |-p_{ji}| \ (\because 0 < c \leq 1) \\ &\leq 1 \ (\because \mathbf{P} \text{ is column-normalized}) \\ &= q_{ii}. \end{aligned} \tag{10}$$

Several properties make the GS method attractive for our problem. First, since $\mathbf{Q}$ is usually a sparse matrix, in Equation 8 the time complexity for $\sum_{\forall k \neq j} q_{jk} s_k$ calculation is sublinear in n. Second, only the non-zero entries of $\mathbf{Q}$ need to be stored. Therefore, it is possible to store all the variables in main memory even for a large networks with thousands to millions of nodes. In addition, the iterative procedure can self-correct the roundoff errors. These advantages allow the GS method to be applied to large scale networks.

To obtain the similarity score between one pair of nodes, Algorithm 1 and Algorithm 2 need to compute the similarity scores between all node pairs in the network, whereas Algorithm 3 only needs to calculate $1/n$ of the network. When an application only requests the similarity scores between few pairs of nodes, the GS method could be much faster.

D. Distributed Gauss-Seidel

Since the original task of calculating $\mathbf{S}$ (Equation 3) can be split into n independent tasks, as shown by Equation 6, the tasks can be assigned to n different machines for parallel computation. We show a MapReduce version of Gauss-Seidel algorithm that can calculate the similarity scores between all pairs of nodes with time complexity $O(n^2/k)$ in Algorithm 4 and Algorithm 5 given k machines ($k \leq n$). The algorithm can be applied on a single machine with k-core as well.

In sum, to get the similarity score between a pair of nodes, the time complexity is lowered from $O(n^2)$ (Algorithm 1) to $O(n)$ (Algorithm 3). To get the similarity score between

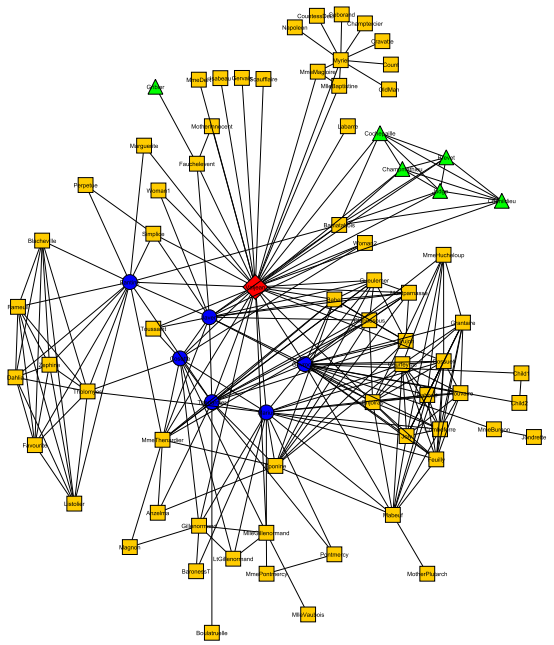


Fig. 2. The *Les Misérables* social network.

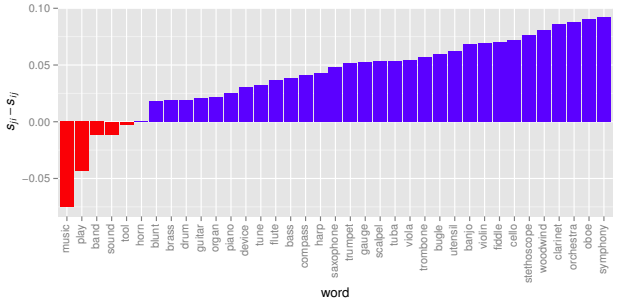


Fig. 3. The score differences of neighbor words of "instrument" to itself.

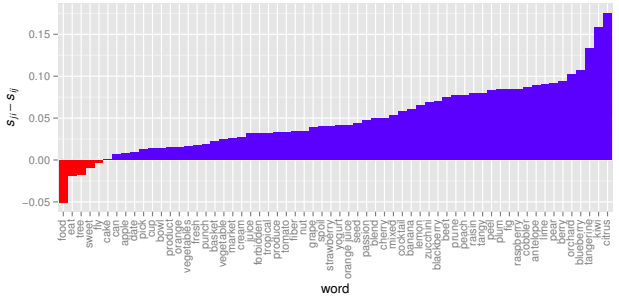


Fig. 4. The score differences of neighbor words of "fruit" to itself.

all pairs of nodes, time complexity is lowered to $O(n^2/k)$ (Algorithm 4 and Algorithm 5). When n machines or n cores are available, the equations can be fully parallelized such that the computation time can further be improved to $O(n)$. The space complexity is reduced from $O(n^2)$ to $O(n)$. Thus, it is feasible to compute ASCOS for large scale complex networks.

Since $\mathbf{Q}$ is usually sparse, Equation 8 can further be parallelized by distributing independent groups of constraints for better efficiency in practice [3].

V. Experiments

A. Similarity Scores of Different Measures

It is difficult to evaluate the similarity measures without conducting extensive user studies [15]. To understand the performance of ASCOS and other measures, we examine the social network of the characters in Victor Hugo's famous novel *Les Misérables*. Each node represents a character in the book, and two nodes are adjacent if they encounter each other [17]. This network is more complex than a toy network but the relationship between the characters is visually understandable. Thus, the outputs of different measures are easier to compare.

A snapshot of the *Les Misérables* social network is shown in Figure 2. The red diamond at the center is Jean Valjean, the main character of *Les Misérables*. The top-6 similar nodes returned by ASCOS and SimRank are highlighted: the green triangles are Valjean's top-6 similar characters returned by SimRank, and the blue circles are Valjean's top-6 similar characters returned by ASCOS. By only observing the network structure, the top similar nodes obtained by ASCOS better fit our intuition. Valjean's most similar character calculated by SimRank is Gribier, who doesn't even directly connect to Valjean. The next five similar characters, Judge, Brevet, Champmathieu, Cochepaille, and Chenildieu, have the same similarity score to Valjean. These characters are thought to be similar with Valjean by SimRank mainly because they are connected to one of Valjean's neighbor Bamatabois, who has only few neighbors except Valjean and the five characters. This is one typical case where SimRank returns counter-intuitive results: given a target node t, SimRank usually regards t's similar nodes as those who can reach t in two steps and the intermediate nodes have few neighbors. As for ASCOS result, Valjean's top-6 similarity characters, Javert, Thenardier, Marius, Gavroche, Cosette, and Fantine, are all directed connected to him. They are also closely connected to other Valjean's neighbors.

To quantitatively measure the performance of ASCOS and other similarity measures, we asked 10 individuals who reported that they know 80% or more of the story of *Les Misérables* to score the returned lists for all methods. Out of them 8 are drum corps performers who played musical songs of *Les Misérables* before, 1 is a graduate student majoring in Comparative Literature, and 1 a graduate student majoring in Computer Science. The scorers were asked to assign 2 points to a character if they believe that the character is highly relevant to Valjean, 1 point if they are somewhat relevant, and 0 if they are minimally relevant. The average points are presented in Table I where the names and average points are highlighted in bold face if the average points are more than 1.5. As shown, SimRank and LHN return characters who are not very relevant to Valjean. Katz performs better than SimRank and LHN. Our proposed ASCOS performs best among these measures: all the top-6 returned characters are highly relevant. The average score of ASCOS is 1.883.

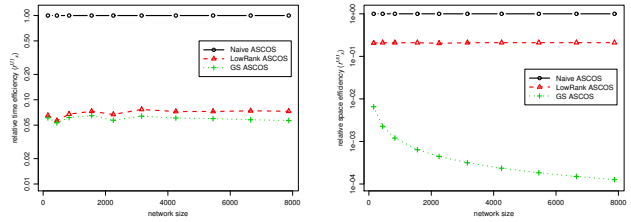
B. Asymmetric Property of ASCOS

As discussed earlier, the ASCOS score s_{ij} from a node i to a node j tends to be smaller than s_{ji} if i is judged to be more salient or general than j. This asymmetric nature makes

TABLE I

A COMPARISON OF JEAN VALJEAN'S TOP SIMILAR CHARACTERS CALCULATED BY ASCOS, SIMRANK, KATZ, AND LHN. THE AVERAGE OF 10 SCORES' GIVEN POINTS ARE LISTED (2: HIGHLY RELEVANT; 1: SOMEWHAT RELEVANT; 0: MINIMALLY RELEVANT). NAMES AND AVERAGE POINTS ARE IN BOLD FACE IF THE AVERAGE VALUES ARE MORE THAN 1.5.

Seq	ASCOS		SimRank		Katz		LHN	
	Name	Avg. Pt.	Name	Avg. Pt.	Name	Avg. Pt.	Name	Avg. Pt.
1	**Javert**	**2.0**	Gribier	0.1	**Javert**	**2.0**	Gervais	0.4
2	**Thenardier**	**1.9**	Judge	0.3	**Gavroche**	**1.6**	MmeDeR	0.3
3	**Marius**	**1.8**	Brevet	0.4	**Thenardier**	**1.9**	Isabeau	0.1
4	**Gavroche**	**1.6**	Champmathieu	0.2	**Marius**	**1.8**	Labarre	0.2
5	**Cosette**	**2.0**	Cochepaille	0.4	Enjolras	1.0	Scaufflaire	0.2
6	**Fantine**	**2.0**	Chenildieu	0.4	Gueulemer	0.2	Marguerite	0.1
Average	-	**1.883**	-	0.3	-	1.417	-	0.217



(a) The run time of different similarity measures to get the similarity score between a pair of nodes for different network sizes.

(b) The required space of different similarity measures to get the similarity score between a pair of nodes under different network size.

Fig. 5. Efficiency comparison

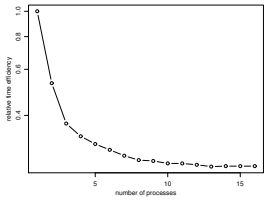


Fig. 6. The relative time efficiency when allocating different number of processes on an 8-core server.

it possible to identify the hierarchical relationship between nodes in a network. To demonstrate this, we utilized the word association norms of over $10,000$ words to generate a word relationship network, in which two words are connected if they are relevant based on a user survey [25]. We illustrate two cases to show the potential of inferring additional semantics between words without using any linguistics.

Figure 3 shows the first case: the ASCOS value difference $s_{ji} - s_{ij}$ given node i is the word "instrument" and node j is one of the 37 neighbor words of the node i. A positive value difference indicates that the word i is likely to be a super-class of the word j. As shown, all the neighbor words representing musical instruments (such as trombone and cello) or other types of instruments (such as compass and stethoscope) have positive value differences. This implies that they are likely to be sub-classes of "instrument". Figure 4 demonstrates another example where node i is the word "fruit" and node j is one of its 62 neighbors. All fruits, such as cherry, peach, and kiwi, are successfully identified as the sub-classes of the word "fruit". However, there are a few words that are not sub-classes of node i, (for example, "symphony" and "orchestra" are not a kind of "instrument", and "cream" and "vegetable" are not one type of "fruit"). But in general most results are reasonable. Since an ASCOS calculation involves no semantic information, the results could be further improved by semantic approaches.

C. Efficiency Comparison

In this section, we empirically compare the run time and space usage of naïve ASCOS with its variations.

Using DBLP Computer Science Bibliography dataset, we construct a coauthorship network between authors who published papers in 1998 in the following conferences: ICDE, ICML, KDD, SIGIR, SIGMOD, VLDB, and WWW. In a similar manner, we construct 9 more coauthorship networks for authors who have publications during 1998-1999, 1998-2000, ..., 1998-2007. Only the giant components of these coauthorship networks are used for analysis.

To empirically compare the efficiency, we measure the run time of naïve ASCOS and all its variations for these 10 different networks with different sizes. We set the run time of naïve ASCOS as the baseline. The relative time efficiency $r_x^{(t)}$ of a variation x is defined as $r_x^{(t)} = t_x / t_{\mathrm{n-ASCOS}}$, where t_x and $t_{\mathrm{n-ASCOS}}$ are the run time of one variation x and the naïve ASCOS respectively. The lower the value, the more efficient the measure is.

Figure 5(a) shows the relative time efficiency of getting the similarity scores between one pair of node for different network sizes. Note that the y axis is logarithm scale. Each point on the Figure is the average of 10 independent trials. As shown, The naïve ASCOS is slower than all its variations. For low rank ASCOS, we set the rank r to be $1/10$ of the total number of nodes. Although r can be set to a smaller number to get a faster run time, such a small number may decrease the accuracy of the approximation. The run time of both low rank ASCOS and GS ASCOS are both around $1/20$ of naïve ASCOS, and GS ASCOS is slightly better than low rank ASCOS consistently.

In a similar manner we define the relative space efficiency $r_x^{(s)}$ of a similarity measure x as $r_x^{(s)} = s_x / s_{\mathrm{n-ASCOS}}$, where s_x and $s_{\mathrm{n-ASCOS}}$ are the required space for getting the similarity score between one pair of nodes with similarity measures x and naïve ASCOS respectively. The result shown in Figure 5(b) shows the GS method is much more memory efficient compared to the rest, especially when the graph size increases. Although the low rank approximation is not as good as GS method, it is still better than naïve ASCOS.

D. Multi-Core/Distributed Computation

Here we empirically compare the run time of the Gauss-Seidel method with and without a distributed computation

environment. Specifically, an 8-core server is used. By varying the requested number of processes from 1 to 16, twice the number of cores, we show their relative time efficiency, which is defined as $r_n = t_{\mathrm{n-processes}}/t_{1-\mathrm{process}}$, where $t_{\mathrm{n-processes}}$ and $t_{1-\mathrm{process}}$ are the run time when requesting n processes and 1 process respectively.

Figure 6 shows the average of 10 independent experiments with y axis in logarithm scale. The relative time performance is roughly the inverse of the number n of requested processes when n is not larger than the total number of CPUs. This suggests that the distributed GS method is highly scalable.

When n is larger than the number of available CPUs, the relative time performance still improves slightly as n grows. This is because the original task is divided into many small sub-tasks such that when a CPU finishes its current job, it can take an unfinished sub-task from the task pool instead of idling and waiting for other CPUs to finish their jobs.

VI. Conclusion and Future Work

We have shown that the popular global structure based similarity measure SimRank and its variations ignore in their calculation the paths of an odd number of lengths. This can generate counter-intuitive similarity scores, as demonstrated by a synthetic toy network and the *Les Misérables* network. Surprisingly, this problem is not discussed in the literature.

We proposed a new Asymmetric Structure COntext Similarity Measure (ASCOS) to address the problem. Theoretically, ASCOS on average utilizes twice the number of paths compared to SimRank and its variations. This extra information can be used in calculating a better similarity score. Empirically, we compared the similarity scores assigned by ASCOS, SimRank, Katz, and LHN. The results by user evaluation showed that our proposed ASCOS yields better performance. The asymmetric feature of ASCOS was shown to have the ability to identify the hierarchical structure of a network. In addition, we proposed several variations of ASCOS, including one distributed method where similarity scores are more efficiently computed. Empirical experiments on DBLP coauthorship network demonstrated these variations are computationally efficient in both time and space.

Future work would test ASCOS on other types of networks. ASCOS could also be applied to vertex similarity based applications and research, such as link prediction, network clustering, and network evolution. As an example the asymmetric property of ASCOS could be applied to coauthorship networks to infer the adviser-advisee between researchers.

Acknowledgments

We gratefully acknowledge partial support by the National Science Foundation and Dow Chemical.

References

[1] E. Acar, D. Dunlavy, and T. Kolda. Link prediction on evolving data using matrix and tensor factorizations. In *IEEE International Conference on Data Mining Workshops*, pages 262–269. IEEE, 2009.

[2] L. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003.

[3] M. Adams. A distributed memory unstructured Gauss-Seidel algorithm for multigrid smoothers. In *Supercomputing, ACM/IEEE 2001 Conference*. IEEE, 2001.

[4] M. Al, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *SDM Workshop on Link Analysis, Counter-terrorism and Security*, 2006.

[5] A. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[6] Y. Cai, M. Zhang, C. Ding, and S. Chakravarthy. Closed form solution of similarity algorithms. In *Proceedings of the 33rd International SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2010.

[7] H.-H. Chen, L. Gou, X. Zhang, and C. Giles. Collabseer: a search engine for collaboration discovery. In *Proceedings of the 11th ACM/IEEE Joint Conference on Digital Libraries. ACM*, 2011.

[8] H.-H. Chen, L. Gou, X. Zhang, and C. Giles. Discovering missing links in networks using vertex similarity measures. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 138–143. ACM, 2012.

[9] H.-H. Chen, L. Gou, X. Zhang, and C. L. Giles. Capturing missing edges in social networks using vertex similarity. In *Proceedings of the sixth international conference on Knowledge capture*. ACM, 2011.

[10] J. Cullum and R. Willoughby. *Lanczos algorithms for large symmetric eigenvalue computations*, volume 1. Society for Industrial Mathematics, 2002.

[11] Y. Dong, Q. Ke, B. Wang, and B. Wu. Link prediction based on local information. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 382–386. IEEE, 2011.

[12] Y. Fujiwara, M. Nakatsuji, T. Yamamuro, H. Shiokawa, and M. Onizuka. Efficient personalized pagerank with accuracy assurance. In *Proceedings of the 18th ACM International Conference on Knowledge Discovery and Data Mining*. ACM, 2012.

[13] G. Golub and C. Van Loan. *Matrix computations*. Johns Hopkins Univ Pr, 1996.

[14] G. He, H. Feng, C. Li, and H. Chen. Parallel simrank computation on large graphs with iterative aggregation. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2010.

[15] G. Jeh and J. Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 538–543. ACM, 2002.

[16] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[17] D. Knuth. *The Stanford GraphBase: a platform for combinatorial computing*. ACM Press, 1993.

[18] E. Leicht, P. Holme, and M. Newman. Vertex similarity in networks. *Physical Review E*, 73(2):026120, 2006.

[19] C. Li, J. Han, G. He, X. Jin, Y. Sun, Y. Yu, and T. Wu. Fast computation of simrank for static and dynamic information networks. In *Proceedings of the 13th International Conference on Extending Database Technology*. ACM, 2010.

[20] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007.

[21] J.-S. Liu and K.-C. Ning. Applying link prediction to ranking candidates for high-level government post. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 145–152. IEEE, 2011.

[22] L. Lu and T. Zhou. Link prediction in complex networks: a survey. *Physica A: Statistical Mechanics and Its Applications*, 390(6):1150–1170, 2011.

[23] A. Maguitman, F. Menczer, F. Erdinc, H. Roinestad, and A. Vespignani. Algorithmic computation and approximation of semantic similarity. *World Wide Web*, 9(4):431–456, 2006.

[24] E. Minkov, W. Cohen, and A. Ng. Contextual search and name disambiguation in email using graphs. In *Proceedings of the 29th International SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34. ACM, 2006.

[25] D. L. Nelson, C. L. McEvoy, and T. A. Schreiber. The university of south florida free association, rhyme, and word fragment norms. *Behavior Research Methods, Instruments, & Computers*, 36(3):402–407, 2004.

[26] M. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104, 2006.

[27] R. Panigrahy, M. Najork, and Y. Xie. How user behavior is related to social affinity. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 2012.

[28] E. Ravasz, A. Somera, D. Mongru, Z. Oltvai, and A. Barabási. Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586), 2002.

[29] G. Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley, 1989.

[30] P.-N. Tan, M. Steinbach, and V. Kumar. Introduction to data mining. *Addison-Wesley*, 2006.

[31] A. Tversky. Features of similarity. *Psychological Review*, 84(4):327, 1977.

[32] W. Xi, E. Fox, W. Fan, B. Zhang, Z. Chen, J. Yan, and D. Zhuang. Simfusion: measuring similarity using unified relationship matrix. In *Proceedings of the 28th Annual International SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2005.

[33] W. Yu, X. Lin, W. Zhang, Y. Zhang, and J. Le. Simfusion+: extending simfusion towards efficient estimation on large and dynamic networks. In *Proceedings of the 35th International SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2012.

[34] P. Zhao, J. Han, and Y. Sun. P-rank: a comprehensive structural similarity measure over information networks. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 553–562. ACM, 2009.

[35] T. Zhou, L. Lu, and Y. Zhang. Predicting missing links via local information. *The European Physical Journal B-Condensed Matter and Complex Systems*, 2009.