© The British Computer Society 2018. All rights reserved.
For permissions, please e-mail: journals.permissions@oup.com
Advance Access publication on 25 April 2018
doi:10.1093/comjnl/bxy040

Discriminative Distance-Based Network Indices with Application to Link Prediction

Mostafa Haghir Chehreghani*, Albert Bifet and Talel Abdessalem

LTCI, Télécom ParisTech, Université Paris-Saclay, 75013 Paris, France
**Corresponding author: mostafa.chehreghani@gmail.com*

In distance-based network indices, the distance between two vertices is measured by the length of shortest paths between them. A shortcoming of this measure is that when it is used in real-world networks, a huge number of vertices may have exactly the same closeness/eccentricity scores. This restricts the applicability of these indices as they cannot distinguish vertices. Furthermore, in many applications, the distance between two vertices not only depends on the length of shortest paths but also on the number of shortest paths between them. In this paper, first we develop a new distance measure, proportional to the length of shortest paths and inversely proportional to the number of shortest paths, that yields discriminative distance-based centrality indices. We present exact and randomized algorithms for computation of the proposed discriminative indices. Then, by performing extensive experiments, we first show that compared with the traditional indices, discriminative indices have usually much more discriminability. Then, we show that our randomized algorithms can very precisely estimate average discriminative path length and average discriminative eccentricity, using only few samples. Then, we show that real-world networks have usually a tiny average discriminative path length, bounded by a constant (e.g. 2). We refer to this property as the *tiny-world property*. Finally, we present a novel *link prediction* method that uses *discriminative distance* to decide which vertices are more likely to form a link in future, and show its superior performance.

Keywords: social network analysis; distance-based network indices; discriminative indices; closeness centrality; eccentricity; average path length; the tiny-world property; link prediction

Received 30 October 2017; revised 22 March 2018; editorial decision 3 April 2018
Handling editor: Zhaolong Ning

1. INTRODUCTION

An important category of network indices is based on the *distance* (the length of the shortest paths) between every two vertices in the network. It includes *closeness centrality*, *average path length*, *vertex eccentricity*, *average graph eccentricity*, etc. Indices in this category have many important applications in different areas. For example, in disease transmission networks, closeness centrality is used to measure vulnerability to disease and infectivity [1]. In routing networks, vertex eccentricity is used to determine vertices that form the periphery of the network and have the largest worst-case response time to any other device [2, 3]. In biological networks, vertices with high eccentricity perceive changes in concentration of their neighbor enzymes or molecules [4].

Using the length of shortest paths as the distance measure has shortcomings. A well-studied shortcoming is that extending it to disconnected graphs (and also directed graphs) is controversial [5–8]. The other—less studied—shortcoming is that by using this measure, a huge number of vertices may find exactly the same closeness/eccentricity score. For instance, Shun [9] recently reported that around 30% of the (connected) vertices of the Yahoo graph have the same non-zero eccentricity score. Our experiments, reported in Section 6.1, reveal that this happens in many real-world graphs. This restricts the applicability of distance-based indices such as closeness and eccentricity, as they cannot distinguish vertices. For example, when closeness or eccentricity is used for the facility location problem [10], they may not be able to

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

distinguish one location among a set of candidate locations. Finally, in many cases, the distance between two vertices not only depends on the length of shortest paths, but also on the number of shortest paths between them. As a simple example, consider a network of locations where edges are roads connecting the locations. In a facility location problem, given two (or more) candidate locations, we want to choose the one which is more accessible from the rest of the network. Then, we may prefer the location which is slightly farther from the rest of the network but has more connections to the location which is closest to the rest of the network. In particular, if two locations have exactly the same distance from the other locations, the one connected to the rest of the network by more roads is preferred.

These observations motivate us to develop a new distance measure between vertices of a graph that yields *more discriminative* centrality notions. Furthermore, it considers both shortest path length and the number of shortest paths. In this paper, our key contributions are as follows:

- We propose new distance-based network indices, including *discriminative closeness*, *discriminative path length*, *discriminative vertex eccentricity* and *average discriminative graph eccentricity*. These indices are proportional to the length of shortest paths and inversely proportional to the number of shortest paths. Our empirical evaluation of these notions reveals an interesting property of real-world networks. While real-world graphs have the **small-world** property which means they have a small average path length bounded by the logarithm of the number of their vertices, they usually have a considerably smaller average discriminative path length, bounded by a constant (e.g. 2). We refer to this property as the **tiny-world** phenomena.
- We present algorithms for exact computation of the proposed discriminative indices. We then develop a randomized algorithm that precisely estimate average discriminative path length (and average discriminative eccentricity) and show that it can give an (ε,δ)-approximation, where $\varepsilon \in \mathbb{R}^+$ and $\delta \in (0,1)$.
- We perform extensive experiments over several real-world networks from different domains. First, we examine *discriminability* of our proposed indices and show that compared with the traditional indices, they are usually much more discriminative.[1] Second, we evaluate the empirical efficiency of our simple randomized algorithm for estimating average discriminative path length and show that it can very precisely estimate average discriminative path length, using only few samples. Third, we show that our simple randomized algorithm for estimating average discriminative eccentricity can generate high-quality results, using only few samples. This has analogy to the case of average eccentricity where a simple randomized algorithm significantly outperforms more advanced techniques [9].
- In order to better motivate the usefulness of our proposed distance measure in real-world applications, we present a novel *link prediction* method that uses discriminative distance to indicate which vertices are more likely to form a link in future. By running extensive experiments over several real-world datasets, we show the superior performance of our method, compared with the well-known existing methods.

The rest of this paper is organized as follows. In Section 2, preliminaries and necessary definitions related to distance-based indices are introduced. A brief overview on related work is given in Section 3. In Section 4, we introduce our discriminative distance-based indices and discuss their extensions and properties. We present exact and approximate algorithms for computing discriminative indices in Section 5. In Section 6, we empirically evaluate discriminability of our indices and the efficiency and accuracy of our randomized algorithms. In Section 7, we present our link prediction method and show its superior performance. Finally, the paper is concluded in Section 8.

2. PRELIMINARIES

In this section, we present definitions and notations widely used in the paper. We assume that the reader is familiar with basic concepts in graph theory. Throughout the paper, G refers to a graph (network). For simplicity, we assume that G is a connected, undirected and loop-free graph without multi-edges. Throughout the paper, we assume that G is an unweighted graph, unless it is explicitly mentioned that G is weighted. $V(G)$ and $E(G)$ refer to the set of vertices and the set of edges of G, respectively. We use n and m to refer to $|V(G)|$ and $|E(G)|$, respectively. We denote the set of neighbors of a vertex v by $\mathcal{N}(v)$.

A *shortest path* (also called a *geodesic path*) between two vertices $v,u \in V(G)$ is a path whose length is minimum, among all paths between v and u. For two vertices $v,u \in V(G)$, we use $d(v,u)$, to denote the *length* (the number of edges) of a shortest path connecting v and u. We denote by $\sigma(v,u)$ the number of shortest paths between v and u. By definition, $d(v,v) = 0$ and $\sigma(v,v) = 0$. We use $\deg(v)$ to denote the degree of vertex v. The *diameter* of G, denoted by $\Delta(G)$, is defined as $\max_{v,u \in V(G)} d(v,u)$. The *radius* of G is defined as $\min_{v \in V(G)} \max_{u \in V(G) \setminus \{v\}} d(v,u)$.

[1] Note that having a *total ordering* of the vertices is not always desirable and by *discriminative indices*, we do not aim to do so. Instead, we want to have a *partial ordering* over a *huge* number of vertices that using traditional distance-based measures, find exactly the same value.

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

Closeness centrality of a vertex $v \in V(G)$ is defined as [11][2]

$$C(v) = \frac{1}{n-1} \sum_{u \in V(G) \setminus \{v\}} d(v, u). \quad (1)$$

Average path length of graph G is defined as [13]

$$APL(G) = \frac{1}{n \times (n-1)} \sum_{v \in V(G)} \sum_{u \in V(G) \setminus \{v\}} d(v, u). \quad (2)$$

Eccentricity of a vertex $v \in V(G)$ is defined as [14, 15][3]

$$E(v) = \frac{1}{n-1} \max_{u \in V(G) \setminus \{v\}} d(v, u). \quad (3)$$

Average eccentricity of graph G is defined as [14, 15]

$$AE(G) = \frac{1}{n \times (n-1)} \sum_{v \in V(G)} \max_{u \in V(G) \setminus \{v\}} d(v, u). \quad (4)$$

Center of a graph is defined as the set of vertices that have the minimum eccentricity. *Periphery* of a graph is defined as the set of vertices that have the maximum eccentricity.

3. RELATED WORK

The widely used distance-based indices are *closeness* centrality, *average path length*, *eccentricity* and *average eccentricity* defined in Section 2. In all these indices, it is required to compute the distance between every pair of vertices. The best algorithm in theory for solving all-pairs shortest paths is based on matrix multiplication [16] and its time complexity is $O(n^{2.3727})$. However, in practice breadth-first search (for unweighted graphs) and Dijkstra's algorithm (for weighted graphs with positive weights) are more efficient. Their time complexities for all vertices are $O(nm)$ and $O(nm + n^2 \log n)$, respectively. In the following, we briefly review exact and inexact algorithms proposed for computing closeness and eccentricity.

3.1. Closeness centrality and average path length

Eppstein and Wang [17] presented a uniform sampling algorithm that with high probability approximates the inverse closeness centrality of all vertices in a weighted graph G within an additive error $\varepsilon\Delta(G)$. Their algorithm requires $O\left(\frac{\log n}{\varepsilon^2}\right)$ samples and spends $O(n \log n + m)$ time to process each one. Brandes and Pich [18] extended this sampler by considering different non-uniform ways of sampling. Cohen *et al.* [19] combined the sampling method with the *pivoting* approach [20, 21], where pivoting is used for the vertices that are far from the given vertex. Olsen *et al.* [22] suggested storing and re-using the intermediate results that are common among different vertices. Okamoto *et al.* [23] presented an algorithm for ranking top k highest closeness centrality vertices that runs in $O((k + n^{\frac{2}{3}} \log^{\frac{1}{3}} n)(n \log n + m))$ time. There are several extensions of closeness centrality for specific networks. Kang *et al.* [11] defined closeness centrality of a vertex v as the (approximate) average distance from v to all other vertices in the graph and proposed algorithms to compute it in MapReduce. Tarkowski *et al.* [24] developed a game-theoretic extension of closeness centrality to networks with community structure.

3.2. Eccentricity and average eccentricity

Dankelmann *et al.* [25] showed that the average eccentricity of a graph is at least $\frac{9n}{4(\deg_m + 1)} + O(1)$, where $\deg_m$ is the minimum degree of the graph. Roditty and Williams [26] developed an algorithm that gives an estimation $\hat{E}(v)$ of the eccentricity of vertex v in an undirected and unweighted graph, such that $\hat{E}(v)$ is bounded as follows: $\frac{2}{3}E(v) \leq \hat{E}(v) \leq \frac{3}{2}E(v)$. Time complexity of this algorithm is $O(m\sqrt{n \log n})$. Takes and Kosters [3] presented an exact eccentricity computation algorithm, based on lower and upper bounds on the eccentricity of each vertex of the graph. They also presented a pruning technique and showed that it can significantly improve upon the standard algorithms. Chechik *et al.* [27] introduced an $O((m \log m)^{3/2})$ time algorithm that gives an estimate $\hat{E}(v)$ of the eccentricity of vertex v in an undirected and weighted graph, such that $\frac{3}{5}E(v) \leq \hat{E}(v) \leq E(v)$. Shun [9] compared shared-memory parallel implementations of several average eccentricity approximation algorithms. He showed that in practice a two-pass simple algorithm significantly outperforms more advanced algorithms such as [26, 27].

4. DISCRIMINATIVE DISTANCE-BASED INDICES

In this section, we present the family of discriminative distance-based indices.

4.1. Indices

The first index is *discriminative closeness* centrality. Similar to closeness centrality, discriminative closeness is based on

[2]The more common definition of *closeness centrality* is as follows [12]: $C(v) = (n-1)/(\sum_{u \in V(G) \setminus \{v\}} d(v,u))$. In this paper, due to consistency with the definitions of the other distance-based indices, we use the definition presented in Equation (1). Note that this change has no effect on the results presented in the paper and they are still valid for the more common definition of closeness.

[3]Again, while the common definition of *eccentricity* does not have the normalization factor $1/n-1$, here in order to have consistent definitions for all the distance-based indices, we add it to Equation (3).

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

the length of shortest paths between different vertices in the graph. However, unlike closeness centrality, discriminative closeness centrality considers the number of shortest paths, too. For a vertex $v \in V(G)$, *discriminative closeness* of v, denoted with $DC(v)$, is formally defined as follows:

$$DC(v) = \frac{1}{n-1} \sum_{u \in V(G) \setminus \{v\}} \frac{d(v,u)}{\sigma(v,u)}. \quad (5)$$

If in the definition of average path length, closeness centrality is replaced by discriminative closeness centrality defined in Equation (5), we get *average discriminative path length* of G, defined as follows:

$$ADPL(G) = \frac{1}{n \times (n-1)} \sum_{v \in V(G)} \sum_{u \in V(G) \setminus \{v\}} \frac{d(v,u)}{\sigma(v,u)}. \quad (6)$$

In a similar way, *discriminative eccentricity* of a vertex $v \in V(G)$, denoted by $DE(v)$, is defined as follows:

$$DE(v) = \frac{1}{n-1} \max_{u \in V(G) \setminus \{v\}} \frac{d(v,u)}{\sigma(v,u)}. \quad (7)$$

Finally, *average discriminative eccentricity* of G is defined as follows:

$$ADE(G) = \frac{1}{n \times (n-1)} \sum_{v \in V(G)} \max_{u \in V(G) \setminus \{v\}} \frac{d(v,u)}{\sigma(v,u)}. \quad (8)$$

All these notions are based on replacing *distance* by *discriminative distance*, defined as follows. For $v,u \in V(G)$, *discriminative distance* between v and u, denoted with $dd(v,u)$, is defined as $\frac{d(v,u)}{\sigma(v,u)}$. We define *discriminative diameter* and *discriminative radius* of G respectively as follows:

$$DD(G) = \max_{v \in V(G)} \max_{u \in V(G) \setminus \{v\}} \frac{d(v,u)}{\sigma(v,u)}, \quad (9)$$

$$DR(G) = \min_{v \in V(G)} \max_{u \in V(G) \setminus \{v\}} \frac{d(v,u)}{\sigma(v,u)}. \quad (10)$$

Finally, we define *discriminative center* of a graph as the set of vertices that have the minimum discriminative eccentricity; and *discriminative periphery* of a graph as the set of vertices that have the maximum discriminative eccentricity.

4.1.1. Generalizations

We can consider two types of generalizations of Equations (5)–(10). In the first generalization, in the denominator of the equations, instead of using the number of shortest paths, we may use the number of a restricted class of shortest paths, e.g. vertex disjoint shortest paths, edge disjoint shortest paths etc. In the second generalization, instead of directly using distances and the number of shortest paths, we may introduce and use functions f and g, defined respectively on the length and the number of shortest paths. Then, by changing the definitions of f and g, we can switch among different distance-based notions. For example, for any two vertices $v,u \in V(G)$, if $f(d(v,u))$ and $g(\sigma(v,u))$ are, respectively, defined as $d(v,u)$ and 1, we will have the traditional distance-based indices introduced in Section 2. If $f(d(v,u))$ and $g(\sigma(v,u))$ are, respectively, defined as $\frac{1}{d(v,u)}$ and 1, we will have *harmonic closeness* centrality [7] defined as follows:

$$HC(v) = \frac{1}{n-1} \sum_{u \in V(G) \setminus \{v\}} \frac{1}{d(v,u)}.$$

Then, someone may define *discriminative harmonic closeness* centrality of vertex v as

$$DHC(v) = \frac{1}{n-1} \sum_{u \in V(G) \setminus \{v\}} \frac{\sigma(v,u)}{d(v,u)}, \quad (11)$$

where $f(d(v,u))$ and $g(\sigma(v,u))$ are, respectively, defined as $\frac{1}{d(v,u)}$ and $\frac{1}{\sigma(v,u)}$.

4.1.2. Connection to the other indices

Path-based indices such as *betweenness centrality* [28, 29] (and its generalizations such as *group betweenness* centrality [30] and *co-betweenness* centrality [31]) consider the number of shortest paths that pass over a vertex. However, betweenness centrality does not consider the shortest path length and it is used as an indicator of the amount of control that a vertex has over shortest paths in the network. Some variations of betweenness centrality, such as *length-scaled betweenness centrality* and *linearly scaled betweenness centrality* [32], are more similar to our proposed notions. However, they still measure the amount of control that a vertex has over shortest paths, but give a weight (which is a function of distance) to the contribution of each shortest path. In our proposed notions, the number of shortest paths passing over a vertex does not always contribute to the centrality of the vertex. Indices such as *Katz centrality* [33] and *personalized PageRank* [34] consider both the length and the number of paths between two vertices. However, there are important differences, too. For example, Katz centrality is proportional to both the length and the number of paths. Furthermore, it considers all paths. This makes it inappropriate for the applications where the concept of shortest paths is essential. This index is mainly used in the analysis of directed acyclic graphs. If in the Katz index of two vertices v and u, denoted by $K(v,u)$, the paths are limited to shortest paths (and the bias constant β is set to 0), we can express it using our generalization of discriminative indices. If this limitation is applied, $K(v,u)$ will be defined as $\alpha^{d(v,u)}\sigma(v,u)$, where α is the *attenuation factor* [33]. Then, this index can be seen as a special

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

case of our generalized discriminative index, where $f(d(v,u))$ is defined as $\alpha^{d(v,u)}$ and $g(\sigma(v,u))$ is defined as $\frac{1}{\sigma(v,u)}$.

The other index that may have some connection to our discriminative indices is *clustering coefficient* [35]. Both clustering coefficient and discriminative indices are sensitive to the local density of the vertices, however, they have different goals. While clustering coefficient aims to directly reflect the local density, discriminative indices aim to take into account the density of different regions of the graph, when computing distances.

4.1.3. *Disconnected or directed graphs*

When the graph is disconnected or directed, it is possible that there is no (shortest) path between vertices v and u. In this case, $d(v,u) = \infty$ and $\sigma(v,u) = 0$, hence, $\frac{d(v,u)}{\sigma(v,u)}$ is undefined. For closeness centrality, when $d(v,u) = \infty$, a first solution is to define $d(v,u)$ as n. The rationale is that in this case $d(v,u)$ is a number greater than any shortest path length. We can use a similar technique for discriminative distance: when there is no path from v to u, we define $d(v,u)$ as n and $\sigma(v,u)$ as 1. This discriminative distance will be greater than the discriminative distance between any two vertices v' and u' that are connected by a path from v' to u'. The second solution suggested for closeness centrality is to use harmonic centrality [7]. As stated in Equation (11), this can be applied to discriminative closeness, too. When $d(v,u) = \infty$, Equation (11) yields $\frac{0}{\infty}$, which is conventional to define as 0.

4.1.4. *A property*

A nice property of shortest path length is that for vertices v,u, $w \in V(G)$ such that w is on a shortest path between v and u, the following holds: $d(v,u) = d(v,w) + d(w,u)$. This property is useful in e.g. designing efficient distance computation algorithms. This property does not hold for discriminative distance as $dd(v,u)$ can be less than or equal to $dd(v,w) + dd(w,u)$. An example is presented in Fig. 1. However, we believe this is not a serious problem. The reason is that more than shortest path length that satisfies the above-mentioned property, discriminative distance is based on the number of shortest paths, which satisfies the following property: $\sigma_w(v,u) = \sigma(v,w) \times \sigma(w,u)$, where $\sigma_w(v,u)$ is the number of shortest paths between v and u that pass over w. As we will discuss in Section 5, these two properties can help us to design efficient algorithms for computing discriminative distance-based indices.

4.1.5. *Why our proposed indices are more discriminative*

It is very difficult to theoretically prove that for general graphs, our proposed indices are always more discriminative than the existing distance-based indices. However, we can still provide arguments explaining why in practice our proposed indices are more discriminative. We here focus on unweighted graphs. Let n be the number of vertices. While the maximum possible shortest path length in a graph is $n - 1$, for real-world networks it is much smaller and it is $\log n$. This means the range of all possible values of distance is narrow and as a result, a vertex may have the same distance from many other vertices. This yields that many vertices may have the same closeness/eccentricity scores. However, the range of all possible values of the number of shortest paths between two vertices is much wider and it varies between 1 and an exponential function of n. For example, on the one hand, in the graph of Figure 2, there are $2^{(n/2)-1}$ shortest paths between s and t and on the other hand, in a tree there is only one shortest path between any two vertices. This wider range yields that when the number of shortest paths is involved, the probability that two vertices have exactly the same score drastically decreases.

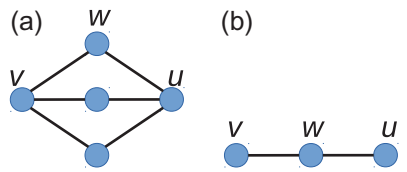

FIGURE 1. In (**a**), we have: $dd(v,u) < dd(v,w) + dd(w,u)$ and, in (**b**), we have: $dd(v,u) = dd(v,w) + dd(w,u)$.

FIGURE 2. There are $2^{(n/2)-1}$ shortest paths between s and t.

4.2. Intuitions

In several cases, the distance between two vertices in the graph not only depends on their shortest path length, but also (inversely) on the number of shortest paths they have. In the following, we discuss some of them.

4.2.1. *Time and reliability of traveling*

A key issue in transportation and logistics [36, 37] and in vehicular social networks (VSNs) [38] is to estimate the *traveling time* and *route reliability* between two points A and B. The time and the reliability of traveling from A to B depend on the structure of the road network and also on the stochastic factors such as weather and traffic incidents. One of the factors that depends on the network structure is the number of ways that someone can travel from A to B. Having several ways to travel from A to B, on the one hand, increases the reliability of traveling. On the other hand, it deceases traffic between A and B and as a result, the traveling time. Therefore, taking into account both the length and the number of shortest paths between A and B (in other words, defining the distance between A and B in terms of both the length and the number of shortest paths) can help to better estimate the time and the reliability of traveling between A and B.

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

4.2.2. *Spread of infections*

It is known that infection and contact rates in a network depend on the community structure of the network and the spread of infections inside a community is faster [39–41]. Consider vertices v_1, v_2 and v_3 such that $d(v_1,v_2) = d(v_1,v_3)$, v_1 and v_2 are in the same community but v_1 and v_3 are not. An infection from v_1 usually spreads to v_2 faster than v_3, or the probability that (after some time steps) v_2 becomes infected by v_1 is higher than the probability that v_3 becomes infected. Here, to describe the distances between the vertices, our discriminative distance measure is a better notion than the shortest path length. Vertices v_1 and v_2 that are inside a community are heavily connected and as a result, they usually have many shortest paths between themselves. In contrast, v_1 and v_3 do not belong to the same community and are not heavily connected, hence, they usually have less shortest paths between themselves. This means $dd(v_1,v_2)$ is smaller than $dd(v_1,v_3)$, which is consistent with the infection rate.

5. ALGORITHMS

In this section, we discuss how discriminative indices can be computed. First in Section 5.1, we present the exact algorithms and then in Section 5.2, we present the approximate algorithms.

5.1. Exact algorithms

In this section, we present the DCC[4] algorithm for computing *discriminative closeness* centrality of all vertices of the network and show how it can be revised to compute the other discriminative indices.

Algorithm 1 High-level pseudo code of the algorithm of computing *discriminative closeness* scores.

```
1:  DCC
2:  Input. A network G.
3:  Output. Discriminative closeness centrality of vertices
    of G.
4:  for all vertex v ∈ V(G) do
5:      I[v] ← 0.
6:  end for
7:  for all vertex v ∈ V(G) do
8:      D,N ← ShortestPathDAG(G,v).
9:      for all vertex u ∈ V(G)\{v} do
10:         I[v] ← I[v] + 1/(n−1) × D[u]/N[u].
11:     end for
12: end for
13: return I.
```

[4]DCC is an abbreviation for **D**iscriminative **C**loseness **C**alculator.

Algorithm 1 shows the high-level pseudo code of the algorithm. DCC is an iterative algorithm where at each iteration, discriminative closeness of a vertex v is computed. This is done by calling the ShortestPathDAG method for v. Inside ShortestPathDAG, the distances and the number of shortest paths between v and all other vertices in the graph are computed. If G is unweighted, this is done by a breadth-first search starting from v. Otherwise, if G is weighted with positive weights, this is done using Dijkstra's algorithm [42]. A detailed description of ShortestPathDAG can be found in several graph theory books, including [43], hence, we here ignore it.

5.1.1. *Computing the other indices*

Algorithm 1 can be revised to compute *average discriminative path length* of G, *discriminative eccentricity* of vertices of G and *average discriminative eccentricity* of G.

- $ADPL(G)$: After Line 12 of Algorithm 1 (where the $I[v]$ values are already computed), $ADPL(G)$ can be computed as $\frac{\sum_{v \in V(G)} I[v]}{n}$.
- $DE(v)$: If Line 10 of Algorithm 1 is replaced by the following lines:

  if $\frac{1}{n-1} \times \frac{D[u]}{N[u]} > I[v]$ **then**

  $I[v] \leftarrow \frac{1}{n-1} \times \frac{D[u]}{N[u]}$.

  end if

  then, the algorithm will compute discriminative eccentricity of the vertices of G and will store them in I.
- $ADE(G)$: After computing *discriminative eccentricity* of all vertices of G and storing them in I, $ADE(G)$ can be computed as $\frac{\sum_{v \in V(G)} I[v]}{n}$.

In a similar way, Algorithm 1 can be revised to compute discriminative diameter and discriminative radius of G.

5.1.2. *Complexity analysis*

For unweighted graphs, each iteration of the loop in Lines 7–12 of method DCC takes $O(m)$ time. For weighted graphs with positive weights, using a Fibonnaci heap, it takes $O(m + n \log n)$ time [43]. This means discriminative closeness centrality and discriminative eccentricity of a given vertex can be computed, respectively, in $O(m)$ time and $O(m + n \log n)$ time for unweighted and weighted graphs with positive weights. However, computing average discriminative path length and/or average eccentricity of the graph requires, respectively, $O(nm)$ time and $O(nm + n^2 \log n)$ time for unweighted graphs and weighted graphs with positive weights. Space complexity of each iteration (and the whole algorithm), for both unweighted graphs and weighted

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

graphs with positive weights, is $O(n + m)$ [43]. Note that these complexities are the same as complexities of computing traditional distance-based indices. The reason is that in addition to computing the distances between v and all other vertices of the graph, ShortestPathDAG can also compute the number of shortest paths, without having any increase in the complexity [43].

5.2. Randomized algorithms

While discriminative closeness and discriminative eccentricity of a given vertex can be computed efficiently, the algorithms of computing average discriminative path length and average discriminative eccentricity of the graph are expensive in practice, even for mid-size networks. This motivates us to present randomized algorithms for ADPL and ADE that can be performed much faster, at the expense of having approximate results.

Algorithm 2 High-level pseudo code of the algorithm of estimating *average discriminative path length*.

```
1:  RandomADPL
2:  Input. A network G and the number of samples T.
3:  Output. Estimated average discriminative path length of G.
4:  β ← 0.
5:  for all t = 1 to T do
6:      Select a vertex v ∈ V(G) uniformly at random.
7:      D,N ← ShortestPathDAG(G,v).
8:      β_t ← 1/(n−1) × Σ_{u∈V(G)\{v}} D[u]/N[u].
9:      β ← β + β_t.
10: end for
11: β ← β/T
12: return β.
```

Algorithm 2 shows the high-level pseudo code of the RandomADPL algorithm, proposed to estimate average discriminative path length. The inputs of the algorithm are the graph G and the number of samples (iterations) T. In each iteration t, the algorithm first chooses a vertex v uniformly at random and calls the ShortestPathDAG method for v and G, to compute distances and the number of shortest paths between v and any other vertex in G. Then, it estimates average discriminative path length of G at iteration t as $\frac{1}{n-1} \times \sum_{u \in V(G) \setminus \{v\}} \frac{d(v,u)}{\sigma(v,u)}$ and stores it in β_t. The average of all β_t values computed during different iterations gives the final estimation β of average discriminative path length. Clearly, for unweighted graphs, time complexity of Algorithm 2 is $O(T \times m)$ and for weighted graphs with positive weights, it is $O(T \times m + T \times n \log n)$. In a way similar to Algorithm 1, Algorithm 2 can be modified to estimate *discriminative eccentricity* of graph G, where the details are omitted.

In the rest of this section, we provide an error bound for our estimation of *average discriminative path length*. First in Proposition 5.1, we prove that in Algorithm 2 the expected value of β is $ADPL(G)$. Then in Proposition 5.2, we provide an error bound for β.

PROPOSITION 5.1. *In Algorithm 2, expected value of β_t's $(1 \leq t \leq T)$ and β is $ADPL(G)$.*

Proof. We have

$$\mathbf{E}[\beta_t] = \sum_{v \in V(G)} \left(\frac{1}{n} \times \frac{\sum_{u \in V(G) \setminus \{v\}} \frac{d(v,u)}{\sigma(v,u)}}{n-1} \right) = ADPL(G),$$

where $\frac{1}{n}$ comes from the uniform distribution used to choose vertices of G. Then, we have: $\mathbf{E}[\beta] = \frac{\sum_{t=1}^{T} \mathbf{E}[\beta_t]}{T} = \frac{T \times \mathbf{E}[\beta_t]}{T} = ADPL(G)$. □

PROPOSITION 5.2. *In Algorithm 2, let G be a connected and undirected graph. For a given $\varepsilon \in \mathbb{R}^+$, we have*

$$\mathbf{P}[|ADPL(G) - \beta| > \varepsilon] \leq 2 \exp\left(-2 \times T \times \left(\frac{\varepsilon}{\Delta(G)} \right)^2 \right). \tag{12}$$

Proof. The proof is done using Hoeffding's inequality [44]. Let $X_1,\ldots,X_n$ be independent random variables bounded by the interval $[a,b]$, i.e. $a \leq X_i \leq b$ $(1 \leq i \leq n)$. Let also $\bar{X} = \frac{1}{n}(X_1 + \cdots + X_n)$. Hoeffding [44] showed that

$$\mathbf{P}[|\mathbf{E}[\bar{X}] - \bar{X}| > \varepsilon] \leq 2 \exp\left(-2n \left(\frac{\varepsilon}{b-a} \right)^2 \right). \tag{13}$$

On the one hand, for any two distinct vertices $v,u \in V(G)$, we have $d(v,u) \leq \Delta(G)$ and $\sigma(v,u) \geq 1$. Therefore, $\frac{d(v,u)}{\sigma(v,u)} \leq \Delta(G)$ and as a result, $\beta_t \leq \Delta(G)$ $(1 \leq t \leq T)$. On the other hand, for any two distinct vertices $v,u \in V(G)$, we have $\frac{d(v,u)}{\sigma(v,u)} > 0$. Therefore, $\beta_t > 0$, for $1 \leq t \leq T$. Note that in Algorithm 2 vertices u are chosen independently and, therefore, variables β_t are independent. Hence, we can use Hoeffding's inequality, where X_i's are β_t's, $\bar{X}$ is β, n is T, a is 0 and b is $\Delta(G)$. Putting these values into Inequality 13 yields Inequality 12. □

Real-world networks have a small diameter, bounded by the logarithm of the number of vertices in the network [35]. This, along with Inequality 12, yields[5]

[5]Note that in Inequality 14, both β and ε are in $\mathbb{R}^+$ and since β and its expected value are not bounded by (0,1) and they are considerably larger than 0 (and they can be larger than 1), ε is usually set to a value much larger than 0 (and even larger than 1, such as $\log n$).

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

$$\mathbf{P}[|ADPL(G) - \beta| > \varepsilon] \leq 2\exp\left(-2 \times T \times \left(\frac{\varepsilon}{\log n}\right)^2\right). \quad (14)$$

Inequality 14 says that for given values $\varepsilon \in \mathbb{R}^+$ and $\delta \in (0,1)$, if T is chosen such that $T \geq \frac{\ln\left(\frac{2}{\delta}\right)(\log n)^2}{2\varepsilon^2}$, Algorithm 2 estimates average discriminative path length of G within an additive error ε with a probability at least $1-\delta$. Our extensive experiments reported in Table 1 of Section 6 (the rightmost column) show that many real-world networks have a very small *discriminative diameter*, much smaller than the logarithm of the number of vertices they have. So, we may assume that their discriminative diameter is bounded by a constant c. For such networks, using only $\frac{c^2 \times \ln\left(\frac{2}{\delta}\right)}{2\varepsilon^2}$ samples, Algorithm 2 can estimate average discriminative path length within an additive error ε with a probability at least $1-\delta$.

6. EXPERIMENTAL RESULTS

We perform extensive experiments on real-world networks to assess the quantitative and qualitative behavior of our proposed algorithms. The programs are compiled by the GNU C++ compiler 5.4.0 using optimization level 3. We do our tests over (largest connected components of) several real-world datasets from different domains, including the *dblp0305*, *dblp0507* and *dblp9202* co-authorship networks [45], the *facebook-uniform* social network [46], the *flickr* network [47], the *gottron-reuters* network [48], the *petster-friendships* network [49], the *pics_ut* network [49], the *web-Stanford* network [50], the *web-NotreDame* network [51], the *citeulike-ut* network [52], the *epinions* network [53] and the *wordnet* network [54]. All the networks are treated as undirected graphs. When a graph is disconnected, we consider only its largest component. Table 1 summarizes specifications of the largest components of our real-world networks.

6.1. Empirical evaluation of discriminability

We measure *discriminability* of a centrality notion in terms of its power in assigning distinguished values to the vertices. Hence, for each centrality notion and over each network G, we define *discriminability* as

$$\frac{\#\text{distinct centrality scores}}{\#\text{vertices of } G} \times 100.$$

Among different distance-based notions studied in this paper, we investigate discriminability of *discriminative closeness* centrality. The reason is that on the one hand, notions such as average discriminative path length and average discriminative eccentricity are graph characteristics, rather than vertex properties. Hence, it does not make sense to measure their discriminability. On the other hand, closeness centrality is a much more common distance-based notion to rank vertices than the other distance-based notions such eccentricity. We compare *discriminative closeness* centrality against *closeness* centrality, as well as a number of centrality notions that are not based on distance, including *betweenness* centrality [28], *length-scaled betweenness* centrality [32] *linearly scaled betweenness* centrality [32] and *Katz* centrality [33].[6] Katz centrality has three parameters to adjust: the damping factor α, the bias constant β and the convergence tolerance *tol*. Similar to e.g. [55], we set β to 1. In order to guarantee convergence, α must be less than the inverse of the largest eigenvalue of the graph. Here we set it to 0.001, which is one of the values used in the experiments of [55]. We set *tol* to $1e-10$.

Table 2 reports the discriminability results. In the table, a 'higher percentage' means a 'higher discriminability' of the centrality notion. The followings can be seen in the table. First, *discriminative closeness centrality* is always more discriminative than the other indices. Second, over datasets such as *dblp0305*, *dblp0507*, *dblp9202*, *facebook-uniform* and *web-Stanford*, discriminability of discriminative closeness centrality is significantly larger than discriminability of the other indices. In fact, when over a network discriminability of the other indices is very low, discriminative closeness centrality becomes significantly more discriminative than them. However, when the other indices are discriminative enough, the difference between discriminability of the indices is less considerable. Third, Katz centrality is usually more discriminative than closeness centrality, betweenness centrality, length-scaled betweenness centrality and linearly scaled betweenness centrality. The only exception is *web-NotreDame*, where length-scaled betweenness centrality and linearly scaled betweenness centrality are more discriminative than Katz centrality. However, unlike the other indices, Katz centrality uses all paths, which makes it improper for the applications where the concept of shortest paths is essential. Fourth, while in most cases, betweenness centrality and its variations are more discriminative than closeness centrality, in a few cases, e.g. over *pics_ut*, *epinions* and *wordnet*, closeness centrality is more discriminative than betweenness centrality and its variations. Fifth, while length scaled and linearly scaled betweenness centrality always show the same discriminability, they slightly improve discriminability of betweenness centrality. However, this improvement is not considerable.

Figure 3 compares running times of computing different indices. Since betweenness centrality, length scaled betweenness centrality and linearly scaled betweenness centrality follow exactly the same procedure and differ only in the way of aggregating the computed scores, we report only one time for

[6] To compute betweenness centrality and its variations, we use boost graph library (http://www.boost.org/doc/libs/1_66_0/libs/graph/doc/index.html) and to compute Katz centrality, we use NetworKit (https://networkit.iti.kit.edu/), where all these algorithms are implemented in C++.

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

TABLE 1. Specifications of the largest component of the real-world datasets.

Dataset	Link	# vertices	# edges	Discriminative diameter
dblp0305	http://www-kdd.isti.cnr.it/GERM/	109 045	233 962	2
dblp0507	http://www-kdd.isti.cnr.it/GERM/	135 116	290 364	2
dblp9202	http://www-kdd.isti.cnr.it/GERM/	129 074	277 082	2
facebook-uniform	http://odysseas.calit2.uci.edu/doku.php/public:online_social_networks	134 304	135 532	2
flickr	http://konect.uni-koblenz.de/networks/flickrEdges	73 342	2 619 711	5
gottron-reuters	http://konect.uni-koblenz.de/networks/gottron-reuters	38 677	978 461	5
petster-friendships	http://konect.uni-koblenz.de/networks/petster-friendships-cat	148 826	5 449 508	8
pics_ut	http://konect.uni-koblenz.de/networks/pics_ut	82 035	2 300 296	5
web-Stanford	http://snap.stanford.edu/data/web-Stanford.html	255 265	2 234 572	16
web-NotreDame	http://snap.stanford.edu/data/web-NotreDame.html	325 729	1 524 589	28
citeulike-ut	http://konect.uni-koblenz.de/networks/citeulike-ut	153 277	2 411 940	7
epinions	http://konect.uni-koblenz.de/networks/epinions	119 130	834 000	15
wordnet	http://konect.uni-koblenz.de/networks/wordnet-words	145 145	656 230	15

TABLE 2. Comparison of *discriminability* of the centrality notions over different real-world networks. For each dataset, the most discriminative index is highlighted in bold.

Database	Discriminative closeness	Closeness	Betweenness	Length scaled betweenness	Linearly scaled betweenness	Katz
dblp0305	**2.7805**	0.0201	0.0403	0.0403	0.0403	0.4447
dblp0507	**2.7013**	0.0155	0.0325	0.0325	0.0325	0.3804
dblp9202	**3.2973**	0.0147	0.0263	0.0263	0.0263	0.3091
facebook-uniform	**5.6178**	0.0446	0.0528	0.0528	0.0528	0.9039
flickr	**92.7694**	4.4435	84.8381	85.0835	85.0835	90.4365
gottron-reuters	**88.9934**	25.8810	74.3956	74.3956	74.3956	88.9753
petster-friendships	**70.0764**	39.2176	65.7049	65.7317	65.7317	70.0260
pics_ut	**50.5113**	36.3552	33.4028	33.5210	33.5210	42.6196
web-Stanford	**97.3376**	18.9258	26.6542	27.2861	27.2861	31.6122
web-NotreDame	**29.9819**	18.5230	18.1245	19.2402	19.2402	19.0277
citeulike-ut	**45.2540**	30.4546	28.6135	28.7185	28.7185	34.3032
epinions	**70.0218**	57.0679	42.1220	45.5745	45.5745	60.1922
wordnet	**58.8907**	51.8770	38.8838	40.5187	40.5187	52.8340

all of them. As can be seen in the figure, since Katz centrality does not find shortest paths and at each vertex, simply follows all its neighbors, it is computed much faster than the other indices. Closeness centrality is computed faster than discriminative closeness and discriminative closeness is computed faster betweenness centrality and its variations. Note that the algorithms of computing closeness centrality and discriminative closeness centrality have the same time complexity. However, to compute discriminative closeness centrality, compared to closeness centrality we require to perform extra operations (e.g. counting the number of shortest paths). This makes it in practice slower than closeness centrality.

6.2. Empirical evaluation of randomized algorithms

Table 3 presents the results of the empirical evaluation of our proposed randomized algorithm for estimating average discriminative path length. When estimating average discriminative path length or average discriminative eccentricity, we define relative error of the approximation algorithm as

$$\frac{|\text{exact score} - \text{approximate score}|}{\text{exact score}} \times 100,$$

where exact score and approximate score are, respectively, the values computed by the exact and approximate algorithms. Sample sizes are expressed in terms of the percentages of the number of vertices of the graph. We examine the algorithm for three sample sizes: 10% of the number of vertices, 1% of the number of vertices and 0.1% of the number of vertices. As can be seen in the table, only a very small sample size, e.g. 0.1% of the number of vertices, is sufficient to have an accurate estimation of average discriminative path length. Over all the datasets, except *gottron-reuters*, this sample size gives a relative error less than 3%. In particular, relative error

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

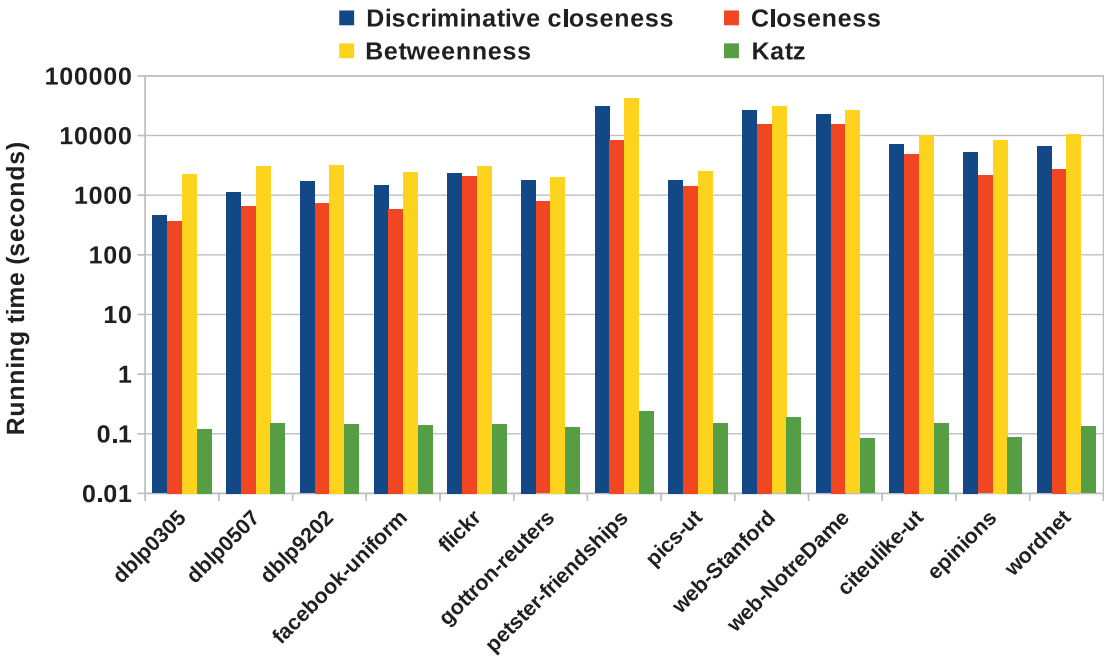

FIGURE 3. Running time of computing discriminative closeness centrality, closeness centrality and betweenness centrality (and its variations). The vertical axis is in logarithmic scale.

in the datasets *dblp0305*, *dblp0507*, *dblp9202* and *facebook-uniform* is very low. This is consistent with our analysis presented in Section 5.2 and is due to very small discriminative diameter of these networks.

Table 3 also compares average discriminative path length of the networks with their average path length. For all the datasets, except *dblp0305*, *dblp0507*, *dblp9202* and *facebook-uniform*, average discriminative path length is considerably smaller than average path length. It may seem surprising that despite very high discriminability of *discriminative closeness* compared to *closeness* over *dblp0305*, *dblp0507*, *dblp9202* and *facebook-uniform*, the differences between *average discriminative path length* and *average path length* are tiny. The reason is that over these datasets, on the one hand, between a huge number of pairs of vertices there is only one shortest path; a few pairs have two shortest paths and only a very tiny percentage of pairs have three or more shortest paths. Therefore, those pairs that have more than one shortest paths do not have a considerable contribution to *ADPL*, hence, *ADPL* and *APL* find very close values. However, on the other hand, for each vertex v there are a different number of vertices to which v is connected by two or more shortest paths. This is sufficient to distinguish its discriminative closeness from the other vertices.

Table 4 reports the results of the empirical evaluation of our randomized algorithm for estimating average discriminative eccentricity. Similar to the case of average discriminative path length, we test the algorithm for three different sample sizes and our experiments show that only a small sample size, e.g. 0.1% of the number of vertices, can yield a very accurate estimation of average discriminative eccentricity. In our experiments, for the sample size 0.1%, relative error is always less than 5%. This high accuracy is due to very small *discriminative diameter* of the networks. Similar to the case of *average eccentricity* where a simple randomized algorithm significantly outperforms advanced techniques [9], our simple algorithms show very good efficiency and accuracy for estimating average discriminative path length and average discriminative eccentricity. Table 4 also shows that similar to *ADPL*, while the datasets *dblp0305*, *dblp0507*, *dblp9202* and *facebook-uniform* have (almost) the same values for *AE* and *ADE*, over the rest of the datasets *ADE* is less than *AE*.

6.3. The tiny-world property

It is well known that in real-world networks, *average path length* is proportional to the logarithm of the number of vertices in the graph and it is considerably smaller than the *largest distance* that two vertices may have in a graph [35]. Our extensive experiments presented in Table 3 reveal that in real-world networks *average discriminative path length* is much more smaller than the *largest discriminative distance*[7] that two vertices may have in a graph and it is bounded by a constant (i.e. 2). This also

[7]Both the *largest distance* and the *largest discriminative distance* that two vertices may have in a graph are equal to the number of vertices in the graph minus 1.

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

implies that *average discriminative path length* of a network is usually considerably smaller than its *average path length*.

This property means that in real-world networks, not only most vertices can be reached from every other vertex by a small number of steps, but also there are many different ways to do so. We call this property the *tiny-world* property. A consequence of this property is that removing several vertices from a real-world network does not have a considerable effect on its *average path length*. Note that this property does not contradict the high discriminability of discriminative closeness; while this property implies that vertices in average have a tiny discriminative distance from each other, the high discriminability of discriminative closeness implies that the discriminative closeness scores of different vertices are less identical than their closeness scores. In Table 3, it can be seen that networks such as *flickr*, *petster-friendships*, *pics_ut* and *citeulike-ut* have an average discriminative path length considerably smaller than the others. This is due to the high density of these networks, which yields that any two vertices may have a shorter distance or more shortest paths.

TABLE 3. Relative error of our randomized average discriminative path length estimation algorithm.

	Exact values		Approximate algorithm	
Database	APL	ADPL	Sample size (%)	Relative error (%)
dblp0305	1.99997	1.99995	10	0.0016
			1	0.0016
			0.1	0.0016
dblp0507	1.99997	1.99996	10	0.0008
			1	0.0008
			0.1	0.0008
dblp9202	1.99997	1.99996	10	0.0015
			1	0.0015
			0.1	0.0012
facebook-uniform	1.99998	1.99997	10	0.0097
			1	0.0099
			0.1	0.0102
flickr	2.3078	0.2787	10	2.5639
			1	0.2887
			0.1	2.7859
gottron-reuters	2.9555	0.6860	10	0.5827
			1	3.6259
			0.1	19.3603
petster-friendships	2.7028	0.2220	10	0.8221
			1	0.4389
			0.1	2.4948
pics_ut	3.6961	0.2953	10	0.4478
			1	1.8667
			0.1	2.9500
web-Stanford	6.8152	0.9509	10	0.6261
			1	1.4910
			0.1	2.7938
web-NotreDame	7.1731	1.5856	10	0.0618
			1	0.6948
			0.1	2.9328
citeulike-ut	3.9376	0.2361	10	0.0355
			1	1.6612
			0.1	2.3498
epinions	4.1814	0.9098	10	0.0240
			1	0.7403
			0.1	0.7527
wordnet	5.5320	1.1141	10	0.1610
			1	0.6710
			0.1	2.7228

7. LINK PREDICTION

In order to better motivate the applicability and usefulness of our proposed distance measure, in this section, we present a novel *link prediction* method, which is based on our new distance measure. We empirically evaluate our method and show that it outperforms the well-known existing link prediction methods.

In the link prediction problem studied in this paper, we are given an unweighted and undirected graph G in which each edge $e = \{u,v\}$ has a timestamp. For a time t, let $G[t]$ denote the subgraph of G consisting of all edges with a timestamp less than or equal to t. Then the link prediction task is defined as follows. Given network $G[t]$ and a time $t' > t$, (partially) sort the list of all pairs of vertices that are not connected in $G[t]$, according to their probability (likelihood) of being connected during the interval $(t,t']$. We refer to the intervals $[0,t]$ and $(t,t']$ as the *training interval* and the *test interval*, respectively.

To generate this (decreasingly) sorted list, existing methods during the training interval compute a similarity matrix S whose entry S_{uv} is the score (probability/likelihood) of having an edge between vertices u and v. Generally, S is symmetric, i.e. $S_{uv} = S_{vu}$. The pairs of the vertices that are at the top of the ordered list are most likely to be connected during the test interval [56]. To compute $S_{u,v}$, several methods have been proposed in the literature, including *the number of common neighbors* [57], *negative of shortest path length* [58] and its variations [59], the *Jaccard's coefficient* [60], the *preferential attachment index* [61], *hitting time* [58], *SimRank* [62], *Katz index* [33], the *Adamic/Adar index* [63] and *resource allocation based on common neighbor interactions* [64]. In the literature, there are also many algorithms that exploit a classification algorithm, with these indices as the features, and try to predict whether a pair of unconnected vertices will be connected during the test interval or not [56, 65, 66].

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

TABLE 4. Relative error of our randomized average discriminative eccentricity estimation algorithm.

	Exact values		Approximate algorithm	
Database	AE (× 1000)	ADE (× 1000)	Sample size (%)	Relative error (%)
dblp0305	0.0183	0.0183	10	0.0013
			1	0.0013
			0.1	0.0013
dblp0507	0.0148	0.0148	10	0.0007
			1	0.0007
			0.1	0.0007
dblp9202	0.0154	0.0154	10	0.0015
			1	0.0015
			0.1	0.0015
facebook-uniform	0.0148	0.0148	10	0.0096
			1	0.0096
			0.1	0.0096
flickr	0.0566	0.0323	10	0.2627
			1	1.1411
			0.1	1.6568
gottron-reuters	0.1159	0.0898	10	0.2327
			1	0.4596
			0.1	4.4685
petster-friendships	0.0432	0.0293	10	0.0749
			1	0.04465
			0.1	2.3459
pics_ut	0.0636	0.0450	10	0.0604
			1	0.7933
			0.1	0.8762
web-Stanford	0.4171	0.0314	10	0.3697
			1	1.0153
			0.1	2.4259
web-NotreDame	0.0852	0.0399	10	0.0235
			1	0.4150
			0.1	0.4150
citeulike-ut	0.0406	0.0262	10	0.3076
			1	0.2016
			0.1	3.2315
epinions	0.0894	0.0676	10	0.0780
			1	0.2170
			0.1	0.3784
wordnet	0.0780	0.0589	10	0.0724
			1	0.4879
			0.1	0.5011

In this section, we propose a new method, called LIDIN,[8] for sorting the list of pairs of unconnected vertices, which is a combination of *shortest path length* and *discriminative distance*. For two pairs of unconnected vertices $\{u_1,v_1\}$ and $\{u_2,v_2\}$, using LIDIN we say vertices u_2 and v_2 are more likely to form a link during the test interval than vertices u_1 and v_1 if

- $d(u_1,v_1) > d(u_2,v_2)$, or
- $d(u_1,v_1) = d(u_2,v_2)$ and $dd(u_1,v_1) > dd(u_2,v_2)$.

The rationale behind LIDIN is that when comparing a pair of vertices u_1,v_1 with another pair u_2,v_2, if $d(u_1,v_1) = d(u_2,v_2)$

[8] LIDIN is an abbreviation for **LI**nk prediction based on **DI**stance and the **N**umber of shortest paths.

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

TABLE 5. Specifications of the temporal real-world datasets used in our experiments for link prediction.

Dataset	Link	#vertices	#temporal edges	Time span
sx-stackoverflow	https://snap.stanford.edu/data/sx-stackoverflow.html	2 601 977	63 497 050	2774 days
sx-mathoverflow	http://snap.stanford.edu/data/sx-mathoverflow.html	24 818	506 550	2350 days
sx-superuser	https://snap.stanford.edu/data/sx-superuser.html	194 085	1 443 339	2773 days
sx-askubuntu	http://snap.stanford.edu/data/sx-askubuntu.html	159 316	964 437	2613 days
wiki-talk-temporal	https://snap.stanford.edu/data/wiki-talk-temporal.html	1 140 149	7 833 140	2320 days
CollegeMsg	http://snap.stanford.edu/data/CollegeMsg.html	1 899	20 296	193 days

but u_2 and v_2 are connected to each other by more shortest paths than u_1 and v_1, then they are more likely to form a link during the test interval. As a special case, for a fixed k, consider the list $L(k)$ consisting of all pairs of unconnected vertices u and v such that $d(u,v) = k$. A network may have many such pairs. It is known that compared to the pairs of unconnected vertices that have distance $k + 1$, members of $L(k)$ are more likely to form a link during the test interval [67, 68]. However, the question remaining open is what elements of $L(k)$ are more likely to be connected than the other members? Using LIDIN, we argue that by increasing the number of shortest paths between the two vertices, the probability of forming a link increases, too.

In order to empirically evaluate this argument, we perform tests over several *temporal* real-world networks, including sx-stackoverflow [69], sx-mathoverflow [69], sx-superuser [69], sx-askubuntu [69], wiki-talk-temporal [69, 70] and CollegeMsg [71]. Table 5 summarizes the specifications of the used temporal real-world datasets. We consider all these networks as simple and undirected graphs, where multi-edges and self-loops are ignored. Since the networks sx-stackoverflow, sx-superuser, sx-askubuntu and wiki-talk-temporal are too large to load their unconnected pairs of vertices in the memory, after sorting their edges based on timestamp, we only consider the subgraphs generated by their first 300 000 edges.

We compare LIDIN with *negative of shortest path length* [58] and the *Adamic/Adar index* [63], denoted, respectively, by -SPL and AA. We choose -SPL because LIDIN is inherently an improvement of -SPL, furthermore, the experiments reported in [66] show that this index outperforms the other topological (global) indices studied in that paper. We choose AA because the experiments reported in [58] show that among 11 studied indices, the Adamic/Adar index has the best relative performance ratio versus random predictions, the best relative performance ratio versus negative of shortest path length predictor and the best relative performance ratio versus common neighbors predictor.[9]

For the graph formed during training interval, we sort (increasingly when LIDIN is used and decreasingly when -SPL and AA are used) the list L of all pairs of unconnected vertices, based on each of the indices.[10] Then, during the test interval, for each edge that connects a pair in L, we examine its rank in L. In order to evaluate the accuracy of a link prediction method, we use two measures *area under the ROC curve* (*AUC*) and *ranking error* (Q). In *AUC*, we measure the probability that a randomly chosen pair of vertices that find a link during the test interval have a higher score than a randomly chosen pair of vertices that do not find a link during the test interval. Formally, *AUC* of a method *ind* is defined as [72]

$$AUC\,(ind) = \frac{n_g + 0.5n_e}{n_t},$$

where n_t is the number of times that we randomly choose two pairs of vertices; one from those that form a link during the test interval and the other from those that do not; n_g is the number of times that the one that forms a link gets a higher score than the other, and n_e is the number of times that the scores of the two chosen pairs are equal. A higher value of *AUC* implies a better link prediction method. In our experiments, we set n_t to the number of edges in the test interval divided by 10. However, similar results can be seen for other values of n_t.

We define the *ranking error* of a method *ind* as

$$Q\,(ind) = \sum_{\{u,v\} \in TE} \frac{rank\,(\{u, v\}, L_{ind})}{|TE|},$$

where L_{ind} is the list L sorted according to *ind*, TE contains those edges of the test interval that connect a pair in L and $rank(\{u,v\},L_{ind})$ returns the rank of $\{u,v\}$ in L_{ind}. For two given indices $ind1$ and $ind2$, $Q(ind1) < Q(ind2)$ means that compared to $ind2$, $ind1$ gives more priority (i.e. a better rank) to the pairs that form a link during the test interval, hence, $ind1$ is a better method than $ind2$.

[9]We can use these indices either as the features of a classification algorithm (like e.g. [66]), or as the criteria of sorting the list of unconnected pairs of vertices (like e.g. [58]). Here, since we want to omit the effect of the classification algorithm and study only the effect of our new notion, we follow the second option.

[10]When any of these indices is used, there might exist two or more pairs that are not sorted by the index. In this case, these pairs are sorted according to the identifiers of the end-points of the edges.

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

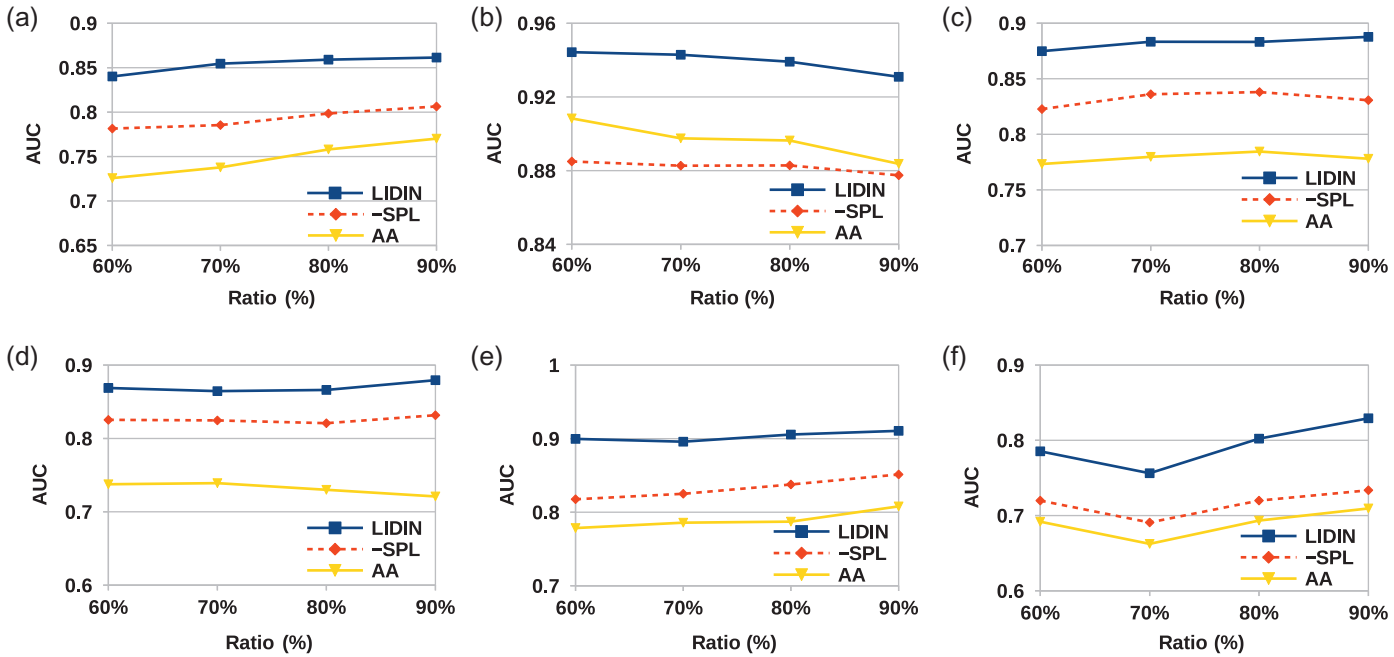

FIGURE 4. *AUC* of different link prediction algorithms. *Ratio* shows the percentage of the edges that form the training interval. (**a**) sx-stackoverow, (**b**) sx-mathoverow, (**c**) sx-superuser, (**d**) sx-askubuntu, (**e**) wiki-talk-temporal and (**f**) CollegeMsg.

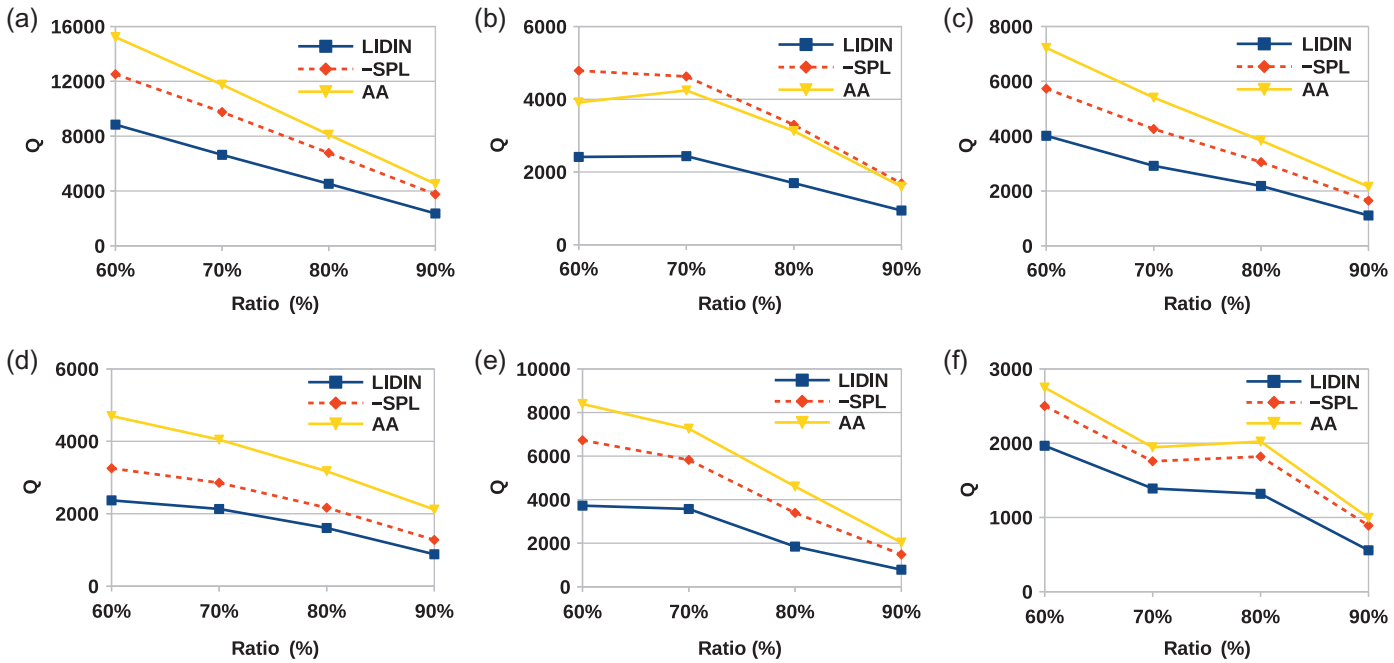

FIGURE 5. The value of Q for different link prediction algorithms. *Ratio* shows the percentage of the edges that form the training interval. (**a**) sx-stackoverow, (**b**) sx-mathoverow, (**c**) sx-superuser, (**d**) sx-askubuntu, (**e**) wiki-talk-temporal and (**f**) CollegeMsg.

We sort the edges of each network according to their timestamps and form the training and test intervals based on the timestamps, i.e. for some given value τ, training interval contains those edges that have a timestamp at most τ and test interval contains those edges that have a timestamp larger than τ. A factor that may affect the empirical behavior of the indices is the value of τ. Therefore and to examine this, we consider four different settings for each network, and choose

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

the values of τ in such a way that training interval includes 60%, 70%, 80% and 90% of the edges. In each case, the rest of the edges which are between a pair of vertices unconnected in the training interval belong to the test interval.

Figure 4 compares *AUC* of different methods. Over all the datasets and in all the settings, LIDIN has the highest *AUC*, therefore, it has the best performance. Figure 5 reports the Q of the studied methods over different datasets. As can be seen in the figure, in all the cases, LIDIN has the lowest Q and hence, the best performance. These tests empirically verify our above-mentioned argument that among all the pairs of unconnected vertices in $L(k)$, those that have a smaller *discriminative distance* (and hence, are closer!), are more likely to form a link. While in most cases -SPL outperforms AA, over *sx-mathoverflow* and for all values of *ratio*, AA has a higher *AUC* and a lower Q than -SPL. The superior performance of our proposed link prediction method suggests that the *inverse of discriminative distance* might be useful in determining similarity between vertices of a network. This means, for example, more than using the fixed criteria for determining the similarity of objects in a Social Internet of Vehicle (SIoV) [73] or friendship of User Equipments [74], someone may also use the inverse of discriminative distance.

8. CONCLUSION AND FUTURE WORK

In this paper, we proposed a new distance measure between vertices of a graph, which is proportional to the length of shortest paths and inversely proportional to the number of shortest paths. We presented exact and randomized algorithms for computation of the proposed discriminative indices and analyzed them. Then, by performing extensive experiments over several real-world networks, we first showed that compared with the traditional indices, discriminative indices have usually much more discriminability. We then showed that our randomized algorithms can very precisely estimate average discriminative path length and average discriminative eccentricity, using only a few samples. In the end, we presented a novel link prediction method, that uses discriminative distance to decide which vertices are more likely to form a link in future, and showed its superior performance compared to the well-known existing measures.

The current work can be extended in several directions. An interesting direction is to investigate distribution of discriminative closeness and discriminative vertex eccentricity in large networks. In particular, it is useful to see whether there exist correlations among discriminative indices on the one hand and other centrality indices such as betweenness and degree on the other hand. The other direction for future work is to develop efficient randomized algorithms for estimating discriminative closeness and discriminative eccentricity of one vertex or a set of vertices and discriminative diameter of the graph. For example, it is interesting to develop algorithms similar to [12] that estimate k highest discriminative closeness scores in the graph. The other extension of the current work is the empirical evaluation of the generalizations of the discriminative indices presented in Section 4. Finally, another extension of the current work is to study discriminative indices of different *network models* [75].

FUNDING

This work has been supported by the ANR project IDOLE.

REFERENCES

[1] Bell, D.C., Atkinson, J.S. and Carlson, J.W. (1999) Centrality measures for disease transmission networks. *Soc. Netw.*, **21**, 1–21.

[2] Magoni, D. and Pansiot, J.J. (2001) Analysis of the autonomous system network topology. *SIGCOMM Comput. Commun. Rev.*, **31**, 26–37.

[3] Takes, F.W. and Kosters, W.A. (2013) Computing the eccentricity distribution of large graphs. *Algorithms*, **6**, 100–118.

[4] Pavlopoulos, G.A., Secrier, M., Moschopoulos, C.N., Soldatos, T.G., Kossida, S., Aerts, J., Schneider, R. and Bagos, P.G. (2011) Using graph theory to analyze biological networks. *BioData Mining*, **4**, 10.

[5] Wasserman, S., Faust, K. and Iacobucci, D. (1994) *Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences)*. Cambridge University Press.

[6] Cornwell, B. (2005) A complement-derived centrality index for disconnected graphs. *Connections*, **26**, 70–81.

[7] Marchiori, M. and Latora, V. (2000) Harmony in the small-world. *Physics A*, **285**, 539–546.

[8] Opsahl, T., Agneessens, F. and Skvoretz, J. (2010) Node centrality in weighted networks: generalizing degree and shortest paths. *Soc. Netw.*, **32**, 245–251.

[9] Shun, J. (2015) An Evaluation of Parallel Eccentricity Estimation Algorithms on Undirected Real-World Graphs. *Proc, 21th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Sydney, NSW, Australia, August 10–13, pp. 1095–1104. ACM, New York, NY, USA.

[10] Koschützki, D., Lehmann, K.A., Peeters, L., Richter, S., Tenfelde-Podehl, D. and Zlotowski, O. (2004) Centrality indices. In Brandes, U. and Erlebach, T. (eds.), *Network Analysis: Methodological Foundations* [outcome of a Dagstuhl seminar, 13–16 April 2004], Lecture Notes in Computer Science, 3418, pp. 16–61. Springer.

[11] Kang, U., Papadimitriou, S., Sun, J. and Tong, H. (2011) Centralities in Large Networks: Algorithms and Observations. *Proc. 11th SIAM Int. Conf. Data Mining, SDM*, Mesa, AZ, USA, April 28–30, pp. 119–130. SIAM, Philadelphia, PA, USA.

[12] Bergamini, E., Borassi, M., Crescenzi, P., Marino, A. and Meyerhenke, H. (2016) Computing Top-k Closeness Centrality Faster in Unweighted Graphs. *Proc. 18th Workshop on Algorithm Engineering and Experiments, ALENEX*, Arlington,

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

VA, USA, January 10, pp. 68–80. SIAM, Philadelphia, PA, USA.

[13] Newman, M.E.J. (2003) The structure and function of complex networks. *SIAM Rev.*, **45**, 167–256.

[14] Chartrand, G., Schultz, M. and Winters, S.J. (1996) On eccentric vertices in graphs. *Networks*, **28**, 181–186.

[15] Husfeldt, T. (2017) Computing Graph Distances Parameterized by Treewidth and Diameter. In Guo, J. and Hermelin, D. (eds.), *11th Int. Symp. Parameterized and Exact Computation (IPEC 2016)*, Aarhus, Denmark, August 24–26, Leibniz International Proceedings in Informatics (LIPIcs), **63**, pp. 16:1–16:11. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany.

[16] Williams, V.V. (2012) Multiplying Matrices Faster than Coppersmith–Winograd. In Karloff, H.J. and Pitassi, T. (eds.), *Proc. 44th Symp. Theory of Computing Conference, STOC 2012*, New York, NY, USA, May 19–22, pp. 887–898. ACM, New York, NY, USA.

[17] Eppstein, D. and Wang, J. (2004) Fast approximation of centrality. *J. Graph Alg. Appl.*, **8**, 39–45.

[18] Brandes, U. and Pich, C. (2007) Centrality estimation in large networks. *Intl. J. Bifurcation Chaos*, **17**, 303–318.

[19] Cohen, E., Delling, D., Pajor, T. and Werneck, R.F. (2014) Computing Classic Closeness Centrality, At Scale. *Proc. 2nd ACM Conf. Online Social Networks*, Dublin, Ireland, October 1–2 COSN '14, pp. 37–50. ACM, New York, NY, USA.

[20] Cohen, E. (2000) Polylog-time and near-linear work approximation scheme for undirected shortest paths. *J. ACM*, **47**, 132–166.

[21] Ullman, J.D. and Yannakakis, M. (1991) High probability parallel transitive-closure algorithms. *SIAM J. Comput.*, **20**, 100–125.

[22] Olsen, P.W., Labouseur, A.G. and Hwang, J. (2014) Efficient Top-k Closeness Centrality Search. In Cruz, I.F., Ferrari, E., Tao, Y., Bertino, E., and Trajcevski, G. (eds.), *IEEE 30th Int. Conf. Data Engineering, ICDE*, Chicago, IL, USA, March 31–April 4, pp. 196–207. IEEE, Washington, DC, USA.

[23] Okamoto, K., Chen, W. and Li, X. (2008) Ranking of Closeness Centrality for Large-Scale Social Networks. In Preparata, F.P., Wu, X., and Yin, J. (eds.), *Frontiers in Algorithmics, Second Annual Int. Workshop, FAW*, Changsha, China, June 19–21, Lecture Notes in Computer Science, **5059**, pp. 186–195. Springer.

[24] Tarkowski, M.K., Szczepanski, P.L., Rahwan, T., Michalak, T.P. and Wooldridge, M. (2016) Closeness Centrality for Networks with Overlapping Community Structure. In Schuurmans, D. and and Wellman, M.P. (eds.), *Proc. 13th AAAI Conf. Artificial Intelligence*, Phoenix, AZ, USA, February 12–17, pp. 622–629. AAAI Press, Palo Alto, CA, USA.

[25] Dankelmann, P., Goddard, W. and Swart, C. (2004) The average eccentricity of a graph and its subgraphs. *Utilitas Math.*, **65**, 41–51.

[26] Roditty, L. and Vassilevska Williams, V. (2013) Fast Approximation Algorithms for the Diameter and Radius of Sparse Graphs. *Proc. 45th Annual ACM Symposium on Theory of Computing*, Palo Alto, CA, USA, June 1–4 STOC 13, pp. 515–524. ACM, New York, NY, USA.

[27] Chechik, S., Larkin, D.H., Roditty, L., Schoenebeck, G., Tarjan, R.E. and Williams, V.V. (2014) Better Approximation Algorithms for the Graph Diameter. *Proc. 25th Annu. ACM-SIAM Symp. Discrete Algorithms*, Portland, OR, January, 5–7 SODA 14, pp. 1041–1052. SIAM, Philadelphia, PA, USA.

[28] Chehreghani, M.H. (2014) An efficient algorithm for approximate betweenness centrality computation. *Comput. J.*, **57**, 1371–1382.

[29] Chehreghani, M.H., Bifet, A. and Abdessalem, T. (2018) Efficient Exact and Approximate Algorithms for Computing Betweenness Centrality in Directed Graphs. *22nd Pacific-Asia Conf. Knowledge Discovery and Data Mining*, Melbourne, Australia, June 3–6 PAKDD **18** 12 pages.

[30] Veremyev, A., Prokopyev, O.A. and Pasiliao, E.L. (2017) Finding groups with maximum betweenness centrality. *Optim. Methods Software*, **32**, 369–399.

[31] Chehreghani, M.H. (2014) Effective Co-betweenness Centrality Computation. *Proc. 7th ACM Int. Conf. Web Search and Data Mining*, New York, NY, USA, February 24–28 WSDM 14, pp. 423–432. ACM, New York, NY, USA.

[32] Brandes, U. (2008) On variants of shortest-path betweenness centrality and their generic computation. *Soc. Netw.*, **30**, 136–145.

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

[34] Haveliwala, T.H. (2002) Topic-Sensitive Pagerank. *Proc. 11th Int. Conf. World Wide Web*, Honolulu, HI, USA, May 07–11 WWW 02, pp. 517–526. ACM, New York, NY, USA.

[35] Watts, D.J. and Strogatz, S.H. (1998) Collective dynamics of small-world'networks. *Nature*, **393**, 409–410.

[36] Polus, A. (1979) A study of travel time and reliability on arterial routes. *Transportation*, **8**, 141–151.

[37] Bonsall, P., Firmin, P., Anderson, M., Palmer, I. and Balmforth, P. (1997) Validating the results of a route choice simulator. *Transport. Res. C Emerg. Technol.*, **5**, 371–387.

[38] Ning, Z., Xia, F., Ullah, N., Kong, X. and Hu, X. (2017) Vehicular social networks: enabling smart mobility. *IEEE Commun. Mag.*, **55**, 16–55.

[39] Ball, F., Mollison, D. and Scalia-Tomba, G. (1997) Epidemics with two levels of mixing. *Ann. Appl. Prob.*, **7**, 46–89.

[40] Pellis, L., Ferguson, N.M. and Fraser, C. (2011) Epidemic growth rate and household reproduction number in communities of households, schools and workplaces. *J. Math. Biol.*, **63**, 691–734.

[41] Liu, Z. and Hu, B. (2005) Epidemic spreading in community networks. *Europhys. Lett.*, **72**, 315.

[42] Dijkstra, E.W. (1959) A note on two problems in connexion with graphs. *Numer. Math.*, **1**, 269–271.

[43] Diestel, R. (2010) *Graph Theory* (4th ed). Springer, Heidelberg.

[44] Hoeffding, W. (1963) Probability inequalities for sums of bounded random variables. *J. Am. Stat. Assoc.*, **58**, 13–30.

[45] Berlingerio, M., Bonchi, F., Bringmann, B. and Gionis, A. (2009) Mining Graph Evolution Rules. *Proc. Eur. Conf. Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, Bled, Slovenia, September 7–11, 2009, Lecture Notes in Artificial Intelligence, 75, pp. 115–130. Springer.

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024

[46] Gjoka, M., Kurant, M., Butts, C.T. and Markopoulou, A. (2010) Walking in Facebook: A Case Study of Unbiased Sampling of OSNS. *Proc. 29th Conf. Information Communications*, San Diego, California, USA, March 14–19 INFOCOM'10, pp. 2498–2506. IEEE Press, Piscataway, NJ, USA.

[47] McAuley, J. and Leskovec, J. (2012) Learning to Discover Social Circles in Ego Networks. *Proc. 25th Int. Conf. Neural Information Processing Systems – Volume 1*, Lake Tahoe, Nevada, December 3–6 NIPS'12, pp. 539–547. Curran Associates Inc., USA.

[48] Lewis, D.D., Yang, Y., Rose, T.G. and Li, F. (2004) RCV1: a new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, **5**, 361–397.

[49] Kunegis, J. (2013) Konect: The Koblenz Network Collection. *Proc. 22nd Int. Conf. World Wide Web*, Rio de Janeiro, Brazil, May 13–17 WWW 13 Companion, pp. 1343–1350. ACM, New York, NY, USA.

[50] Leskovec, J., Lang, K., Dasgupta, A. and Mahoney, M. (2009) Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. *Internet Math.*, **6**, 29–123.

[51] Albert, R., Jeong, H. and Barabasi, A.L. (1999) The diameter of the world wide web. *Nature*, **401**, 130–131.

[52] Emamy, K. and Cameron, R. (2007) Citeulike: a researcher's social bookmarking service. *Ariadne*, **51**.

[53] Massa, P. and Avesani, P. (2005) Controversial Users Demand Local Trust Metrics: An Experimental Study on epinions.com Community. *Proc. 20th National Conf. Artificial Intelligence – Volume 1*, Pittsburgh, PA, July 9–13 AAAI'05, pp. 121–126. AAAI Press, USA.

[54] Fellbaum, C. (ed.) (1998) *WordNet: An Electronic Lexical Database*. MIT Press.

[55] Zhan, J., Gurung, S. and Parsa, S.P.K. (2017) Identification of top-k nodes in large networks using katz centrality. *J. Big Data*, **4**, 16.

[56] Martínez, V., Berzal, F. and Cubero, J.-C. (2016) A survey of link prediction in complex networks. *ACM Comput. Surv.*, **49**, 69:1–69:33.

[57] Newman, M. (2001) Clustering and preferential attachment in growing networks. *Phys. Rev. E*, **64**, 025102.

[58] Liben-Nowell, D. and Kleinberg, J. (2007) The link-prediction problem for social networks. *J. Am. Soc. Inf. Sci. Technol.*, **58**, 1019–1031.

[59] Lebedev, A., Lee, J., Rivera, V. and Mazzara, M. (2017) Link prediction using top-k shortest distances. In Cal, A., Wood, P.T., Martin, N.J., and Poulovassilis, A. (eds.), *Data Analytics—31st British Int. Conf. Databases, BICOD*, London, UK, July 10–12, Lecture Notes in Computer Science, 10365, pp. 101–105. Springer.

[60] Salton, G. and McGill, M.J. (1986) *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc, New York, NY, USA.

[61] Barabasi, A., Jeong, H., Neda, Z., Ravasz, E., Schubert, A. and Vicsek, T. (2002) Evolution of the social network of scientific collaborations. *Physics A*, **311**, 590–614.

[62] Jeh, G. and Widom, J. (2002) SimRank: A Measure of Structural-Context Similarity. *Proc. 8th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada, July 23–26 KDD 02, pp. 538–543. ACM, New York, NY, USA.

[63] Adamic, L.A. and Adar, E. (2001) Friends and neighbors on the web. *Soc. Netw.*, **25**, 211–230.

[64] Zhang, J., Zhang, Y., Yang, H. and Yang, J. (2014) A link prediction algorithm based on socialized semi-local information. *J. Comput. Inform. Syst.*, **10**, 4459–4466.

[65] Lu, L. and Zhou, T. (2011) Link prediction in complex networks: a survey. *Physics A*, **390**, 1150–1170.

[66] Hasan, M.A., Chaoji, V., Salem, S. and Zaki, M. (2006) Link Prediction using Supervised Learning. *Proc. SDM Workshop on Link Analysis, Counterterrorism and Security*, Bethesda, MD, USA, April, 20–22 10 pages. SIAM, Philadelphia, PA.

[67] Pasta, M.Q., Zaidi, F. and Rozenblat, C. (2014) Generating online social networks based on socio-demographic attributes. *J. Complex Netw.*, **2**, 475–494.

[68] Chehreghani, M.H. and Chehreghani, M.H. (2016) Modeling Transitivity in Complex Networks. In Ihler, A.T. and Janzing, D. (eds.), *Proc. 32nd Conf. Uncertainty in Artificial Intelligence, UAI*, New York, NY, USA, June 25–29. AUAI.

[69] Paranjape, A., Benson, A.R. and Leskovec, J. (2017) Motifs in Temporal Networks. In de Rijke, M., Shokouhi, M., Tomkins, A., and Zhang, M. (eds.), *Proc. 10th ACM Int. Conf. Web Search and Data Mining, WSDM*, Cambridge, United Kingdom, February 6–10, pp. 601–610. ACM, New York, NY, USA.

[70] Leskovec, J., Huttenlocher, D.P. and Kleinberg, J.M. (2010) Governance in Social Media: A Case Study of the Wikipedia Promotion Process. In Cohen, W.W. and Gosling, S. (eds.), *Proc. 4th Int. Conf. Weblogs and Social Media, ICWSM*, Washington, DC, USA, May 23–26. AAAI Press, USA.

[71] Panzarasa, P., Opsahl, T. and Carley, K.M. (2009) Patterns and dynamics of users' behavior and interaction: network analysis of an online community. *JASIST*, **60**, 911–932.

[72] Zhang, P., Wang, X., Wang, F., Zeng, A. and Xiao, J. (2016) Measuring the robustness of link prediction algorithms under noisy environment. *Sci. Rep.*, **6**, 475–494.

[73] Ning, Z., Hu, X., Chen, Z., Zhou, M., Hu, B., Cheng, J. and Obaidat, M.S. (2017) A cooperative quality-aware service access system for social Internet of vehicles. *IEEE Internet Things J.*, **PP**, 1–1.

[74] Ning, Z., Wang, X., Kong, X. and Hou, W. (2017) A social-aware group formation framework for information diffusion in narrowband internet of things. *IEEE Internet Things J.*, **PP**, 1–1.

[75] Chehreghani, M.H. and Abdessalem, T. (2017) Upper and lower bounds for the q-entropy of network models with application to network model selection. *Inform. Process. Lett.*, **119**, 1–8.

Downloaded from https://academic.oup.com/comjnl/article/61/7/998/4985553 by guest on 03 April 2024