Indexing Network Structure with Shortest-Path Trees

MARC MAIER, MATTHEW RATTIGAN, and DAVID JENSEN
University of Massachusetts, Amherst

15

The ability to discover low-cost paths in networks has practical consequences for knowledge discovery and social network analysis tasks. Many analytic techniques for networks require finding low-cost paths, but exact methods for search become prohibitive for large networks, and data sets are steadily increasing in size. Short paths can be found efficiently by utilizing an index of network structure, which estimates network distances and enables rapid discovery of short paths. Through experiments on synthetic networks, we demonstrate that one such novel network structure index based on the shortest-path tree outperforms other previously proposed indices. We also show that it generalizes across arbitrarily weighted networks of various structures and densities, provides accurate estimates of distance, and has efficient time and space complexity. We present results on real data sets for several applications, including navigation, diameter estimation, centrality computation, and clustering—all made efficient by virtue of the network structure index.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data mining*; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods*

General Terms: Algorithms, Measurement

Additional Key Words and Phrases: Knowledge discovery in graphs, network structure index, social network analysis, weighted networks

ACM Reference Format:
Maier, M., Rattigan, M., and Jensen, D. 2011. Indexing network structure with shortest-path trees. ACM Trans. Knowl. Discov. Data 5, 3, Article 15 (August 2011), 25 pages.
DOI = 10.1145/1993077.1993079 http://doi.acm.org/10.1145/1993077.1993079

1. INTRODUCTION

Many important knowledge discovery and social network analysis tasks involve computing network properties and statistics. Graph diameter is an important measure for various applications (e.g., communication or transportation networks), as it provides a guaranteed upper bound on the path length between any two nodes. Closeness centrality [Freeman 1979] provides a measure of the proximity of a node to all other nodes in a network. Closeness centrality is highly useful for studies of information

This research is supported by Lawrence Livermore National Laboratory and the Department of Energy under contract number W7405-ENG-48 and the National Science Foundation under contract number IIS-0326249.
The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily as representing the official policies or endorsements, either expressed or implied, of NSF, LLNL/DOE, or the U.S. Government.
Authors' address: M. Maier, M. Rattigan, and D. Jensen, Computer Science Department, University of Massachusetts, Amherst, MA 01003; email: maier.marc@gmail.com.

DOI 10.1145/1993077.1993079 http://doi.acm.org/10.1145/1993077.1993079

propagation in which a message must quickly reach all nodes in a network. Betweenness centrality is a measure commonly found in social network analysis as it can identify the most critical or vulnerable nodes in a network. Clustering is another fundamental task in knowledge discovery and machine learning. In relational data, instances are connected by a link structure, and clustering becomes a means for detecting community structure within networks. Graph clustering is studied in many fields of research, including physics [Fortunato 2010; Newman 2004b], social network analysis [Wasserman and Faust 1994], and computer science [Flake et al. 2004].

All of these network properties and applications have a common underlying computational model—they all require discovering the shortest, or lowest-cost, paths in a network. However, data sets are typically large, with the number of nodes in the thousands or millions. Standard methods such as Dijkstra's algorithm [Dijkstra 1959] become intractable if large numbers of searches must be performed, which is precisely the case for computing closeness and betweenness centrality, finding a graph's diameter, and certain graph clustering techniques.

A common approach to improving the speed of calculations is to use an index—an auxiliary data structure that provides fast look-ups for common operations. For example, a database index can help queries avoid repeated access to the records in a database. Similarly, a search engine index ensures that a query does not need to scan all documents in a given corpus.

We employ a similar strategy to aid social network analysis by using an index to avoid computationally expensive searches in a graph. The network structure index, or NSI, was recently introduced as a general technique to support efficient distance estimates of nodes in a graph [Rattigan et al. 2006]. The NSI has been shown to work well for applications that require searching in unweighted graphs, such as clustering [Rattigan et al. 2007]. Since NSIs can provide efficient and accurate graph distance estimates, they can enable fast approximation of short paths, which in turn allows for efficient analysis of a network. As with indices for databases and information retrieval, different approaches to indexing have different advantages and disadvantages in terms of applicability, performance, and complexity. In this article, we present a network structure index that is (1) general in its applicability to various network structures with arbitrary edge weights, (2) simple to specify since it involves a single parameter, (3) accurate in its estimates of network distances, and (4) efficient in terms of both time and space complexity.

We introduce a novel approach to indexing based on shortest-path trees, which allows for highly effective analysis of graphs with arbitrary edge weights, link structure, and density. The shortest-path tree [Dijkstra 1959] is a subgraph of the network consisting of the actual shortest paths from a given root node to all other nodes in the network. Intuitively, the shortest-path tree encapsulates part of the actual topology of the network, which is useful for giving accurate distance estimates. This subsequently makes for efficient navigation in the network. As a result, analytic techniques that rely on short paths become feasible.

This article investigates how an NSI based on the shortest-path tree compares to other NSIs from the networking literature [Mao and Saul 2004; Ng and Zhang 2002] and the knowledge discovery community [Rattigan et al. 2006]. Previous research provides a solid framework for the network structure index as a general tool; however, none of the proposed NSIs can scale well in practice and are limited to unweighted networks. Analysis of large networks typically focuses primarily on link structure, but treating the network as composed of weighted edges can enhance network analysis [Newman 2004a]. Weights on edges can characterize the strength of relationships among entities or some other property, such as latency in the Internet or flow in transportation networks (e.g., number of available passenger seats between airports [Barrat

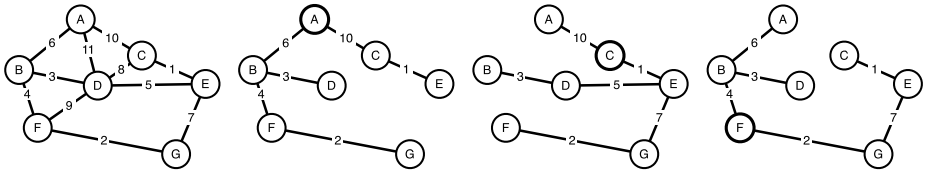


Fig. 1. An example weighted graph and the shortest-path trees rooted at nodes A, C, and F.

et al. 2004]). Consequently, it is crucial to have a network structure index with the flexibility to support networks with arbitrary edge weights.

Through experiments on synthetic networks, we show that the shortest-path tree NSI outperforms other indices in terms of its applicability to arbitrarily weighted networks of various structures and densities and its ability to provide highly accurate estimates of distance. We also characterize the time and space complexity of the NSIs and show that the shortest-path tree NSI is efficient in its construction and storage. Finally, we present results on real and synthetic data sets for several applications, including navigation, diameter estimation, centrality computation, and clustering.

2. NETWORK STRUCTURE INDICES

A network structure index provides distance estimates for any arbitrary pair of nodes in a graph. Formally, NSIs are used to solve the following problem. Given an undirected graph $G = (V, E, w)$ with $|V| = n$ vertices, $|E| = m$ edges, and edge weight function $w : E \rightarrow \Re^+$, provide a distance estimation function $\hat{d} : V \times V \rightarrow \Re^+$ such that $\hat{d} \approx d$, the actual distance function over G. The distance $d(u, v)$ is the minimum cost over all paths between nodes u and v, where cost is the sum of edge weights along a path. Also note that this definition encompasses unweighted graphs. An unweighted graph uses $w : E \rightarrow 1$ to assign uniform edge weights. A good network structure index should provide accurate distance estimates and have low complexity in construction and deployment.

2.1 The Shortest-Path Tree NSI

We propose an NSI that exploits actual distances and the structure of a network. The basic structure that we use is the shortest-path tree (SPT), which comprises the union of all lowest-cost paths from a given root node to all other nodes in the network (see, for example, the graph and three shortest-path trees rooted at nodes A, C, and F depicted in Figure 1). An SPT can be derived from the information obtained by running a single instance of Dijkstra's single-source shortest-path algorithm [Dijkstra 1959]. The intuition behind using this structure as the basis for an NSI is twofold. (1) Since the SPT connects all nodes in a graph, it can provide a distance estimate by simply using the actual path, following the edges within it; and (2) these paths should have low cost since the structure is formed using actual least-cost paths. These two factors suggest that the distance estimates would approximate actual lowest-cost distances. To minimize error, we compute multiple shortest-path trees rooted at randomly selected nodes in the network. Typically, only a few trees are necessary to achieve high performance (see Section 3).

The construction of the SHORTEST-PATH TREE NSI uses the following steps.

(1) Randomly choose a small set of nodes, L.
(2) For $u \in L$, run Dijkstra's algorithm with u as the root and retrieve the shortest-path tree.

(3) Process the tree in order to achieve constant-time lowest common ancestor queries.
(4) Compute and store the distance of the root to each node in the network along the tree.

This NSI consists of distances for each node to each root along with metainformation that allows efficient lowest common ancestor queries. A rooted tree can be preprocessed in linear time with linear space requirements to provide constant-time queries, as discovered by Harel and Tarjan [1984]. Their techniques are complex, so we implement a simpler algorithm with the same complexity requirements that was recently proposed by Bender and Colton [2000]. Given a rooted tree, we perform an Euler tour and store the following information in three arrays: the order in which the nodes appear in the Euler tour, the unweighted distance for each of these nodes to the root, and a representative index for each node based on its first occurrence in the Euler tour. Then, we process these arrays to allow constant-time range minimum queries (find the index of the smallest element in a given range of an array). This can be done by partitioning the arrays and using a hierarchical lookup table. The range minimum query is equivalent to the lowest common ancestor query, given the three arrays. For further details, see Bender and Colton [2000].

The distance function for the SPT NSI returns an actual distance along a shortest-path tree by adding the distance to the root for each node and subtracting twice the distance to the root of the lowest common ancestor of the two nodes. The last subtraction is necessary in the event that the two nodes belong to the same subtree of the SPT. This is computed over each SPT, and the minimum distance is returned. The distance function is formally written as

$$d_{SPT}(u, v) = \min_i (d_i(u, r_i) + d_i(v, r_i) - 2 \cdot d_i(u \cap v, r_i)),$$

where $u, v \in V$ are two arbitrary nodes, and r_i is the root of the ith tree. The lowest common ancestor node is written as $u \cap v$. For example, suppose we request the distance estimate between nodes D and G in Figure 1. First, we would compute the distances along the paths in each of the three shortest-path trees ($SPT_A = 9$, $SPT_C = 12$, and $SPT_F = 9$) and return the minimum, which, in this case, is the actual least-cost distance. In a sense, one can view this approach as a landmarks scheme [Chow 2004; Goldberg and Harrelson 2005] in which, for each tree, a single static landmark (the root node) and one dynamic landmark (the intersection node) are used for the two arbitrary nodes required by the distance function.

The SPT NSI is efficient to construct. A single run of Dijkstra's algorithm requires $O(m \log n)$ using a simple priority queue, and this is repeated l times, once for each node in L. The processing step that enables constant-time lowest common ancestor queries has been shown to take linear time and space. Finally, to compute distances to the root for each node, we simply flood the shortest-path tree, which also requires linear time. Since l is typically chosen to be a small constant (three in the experiments in Section 3), the entire algorithm takes the one-time cost of time $O(m \log n)$ for construction, uses $O(n)$ space, and requires constant time (scaled by l) for distance estimation queries.

2.2 Other NSIs

Previous research [Rattigan et al. 2006] has focused on the use of NSIs in unweighted, undirected networks. The DISTANCE TO ZONE (DTZ) NSI partitions the graph into z mutually exclusive and exhaustive regions, or zones, for each of d dimensions. For each node, it stores the distance to the closest node in each zone. DTZ benefits from accurate distance estimates, but storing and computing this information drastically increases

its construction time and space complexity. The distance function is the maximum distance to either node's respective zone across all dimensions. More formally,

$$d_{DTZ}(u, v) = \max_i \left(d_i(u, \text{zone}(v)), d_i(v, \text{zone}(u))\right).$$

Since DTZ was found to be the best approach at the time for arbitrary unweighted networks, we include it for comparison. However, conceptually, this NSI should not perform well on weighted networks because it exploits random floodings of the graph, which implicitly assumes uniformity of edges.

A large amount of networking research has been dedicated to developing techniques that encapsulate the delay (or latency) space of the Internet, which is an inherently weighted network. Note that these systems have the benefit of being distributed because the nature of the Internet requires researchers to develop techniques with this property. However, this is not an advantage in the current context since analytic tasks are performed locally. We compare the SPT NSI to two additional candidate systems from the networking literature.

The first technique we examine is GLOBAL NETWORK POSITIONING (GNP), by Ng and Zhang [2002]. This system designates several nodes as landmarks and embeds them in a Euclidean space using simplex optimization. This forms a vector space in which all other nodes can be embedded by determining coordinates using the landmark coordinate space as its reference frame. GNP provides a distance function using standard Euclidean distance.

$$d_{GNP}(u, v) = \sqrt{\sum_i (u_i - v_i)^2}.$$

Another approach, developed by Mao and Saul [2004], is based on matrix factorization. Their overall system is known as Internet Distance Estimation Service, but we will simply refer to it as MATRIX FACTORIZATION (MF). The central innovation of this work is that a distance matrix, D, can be approximated by the product of two smaller matrices using matrix factorization. MF works by selecting a small number, l, of landmark nodes, calculating the $l \times l$ interlandmark distance matrix, D, and decomposing it into two smaller matrices, X and Y. Then, for each node u in the network, two vectors are computed: $X_u = d_u Y(Y^T Y)^{-1}$ and $Y_u = d_u X(X^T X)^{-1}$, where d_u is the $1 \times l$ vector of distances to each landmark. This process can provide asymmetric distance estimates to and from the landmark nodes; however, the present work only focuses on undirected graphs, so these vectors are identical. Using these vectors, the distance function is

$$d_{MF}(u, v) = X_u \cdot Y_v^T.$$

An advantage of using MF is that its estimates do not violate the triangle inequality and it works for asymmetric distances. Although these are important factors when considering extensions to the current work, we leave directed graphs and asymmetry for future work. There are numerous other techniques (e.g., VIVALDI [Dabek et al. 2004], BIG-BANG SIMULATION [Shavitt and Tankel 2004], and VIRTUAL LANDMARKS [Tang and Crovella 2003]) that will not be evaluated in this work because their performance has been shown to be comparable to GNP and MATRIX FACTORIZATION [Lua et al. 2005].

2.3 Complexity Comparison

Before comparing the performance of these NSIs, we examine the algorithms themselves. Table I lists the construction time and space complexity of the four NSIs we

Table I. Construction Time and Space Complexity of Four NSIs, Where n is the Number of Nodes, m is the Number of Edges, l is the Number of Landmarks, and $S(\cdot)$ is the Cost of Simplex Optimization

NSI	Time	Space
SPT	$O(l \cdot (m \log n))$	$O(nl)$
GNP	$O(n \cdot S(l))$	$O(nl)$
MF	$O(ml + nl^3)$	$O(nl)$
DTZ	$O(mzd)$	$O(nzd)$

examined. GNP and DTZ have the worst time complexities. The standard criticism against GLOBAL NETWORK POSITIONING is that simplex optimization is expensive and dependent on the choice of initial seeds. DISTANCE TO ZONE comprises simple computations—they are just repeated many times. MATRIX FACTORIZATION requires l floods of the graph to retrieve distances to all landmarks plus the additional cost of computing the annotation vectors for each node, which amounts to matrix multiplication of $O(l)$ matrices. The SHORTEST-PATH TREE NSI is dominated by the cost of running Dijkstra's algorithm (constructing the shortest-path tree). Fortunately, this is only an $O(m \log n)$ operation, and for many real-world networks m is well below $O(n^2)$. For the 99 networks listed in Section A.2 in a large survey of known network data sets [Leskovec 2008], we computed the exponent in the equation $m = n^x$ and found that the median was $x = 1.102$, and in 95% of the cases $x \leq 1.275$.

Since the distance to each zone for each node in each dimension must be stored, the space complexity of DTZ is impractical. If $z \cdot d$ approaches $O(n^2)$, then the NSI would store as much as all-pairs shortest paths. The other three NSIs are similar to each other in terms of space requirements, and since the number of landmarks or trees are typically chosen to be small, each has essentially linear space requirements.

3. EXPERIMENTS

3.1 Performance Comparison

The basic task of interest for NSI performance is navigation. For all experiments, we coupled the distance function of each NSI with A* search [Russell and Norvig 1995]. The distance function thus serves as the heuristic for estimated distance from the target. As the heuristic function more accurately estimates distance, A* search becomes more focused, thus exploring fewer nodes. None of the NSIs presented in this work yield admissible heuristic functions, so A* will not necessarily find optimal paths. However, we evaluated other search methods, including best-first search, and found A* discovered the shortest paths.

As a baseline for evaluating the quality of performance, we compare each NSI with A* to uniform-cost search, a standard search algorithm that finds lowest-cost paths in arbitrary graphs [Russell and Norvig 1995]. Uniform-cost search is a special case of A*, where the heuristic function is set to a constant.

We define three measures, *path ratio*, *exploration ratio*, and *time ratio*, each of which emphasizes a different aspect of search. Each measure is defined with respect to a set of searches S. Path ratio is the sum of the costs of the paths discovered using the search algorithm divided by the sum of the costs of the actual shortest paths in S. Formally, path ratio is defined as

$$P = \frac{\sum_{i \in S} c_f(i)}{\sum_{i \in S} c_o(i)},$$

where $c_f(i)$ is the cost of the found path, and $c_o(i)$ is the cost of the observed shortest path. This quantity is akin to the notion of average stretch in compact routing [Krioukov and Yang 2004]. The optimal value of P is 1.0.

The exploration ratio is defined as the total number of nodes explored by the search algorithm divided by the number of nodes explored using the standard search method—here uniform-cost search. More formally,

$$E = \frac{\sum_{i \in S} e_f(i)}{\sum_{i \in S} e_o(i)}.$$

This measure characterizes the complexity of the search routine and, indirectly, how well the NSI approximates distance. In general, the more accurate the NSI, the fewer the nodes examined to find the target. Values approaching zero are indicative of NSI utility; values greater than 1.0 stress the difficulty in searching, given the NSI distance estimates.

The time ratio measure is the ratio of wall clock time for the search routine using the NSI to our standard baseline search. We define this formally as

$$T = \frac{\sum_{i \in S} t_f(i)}{\sum_{i \in S} t_o(i)}.$$

This measure magnifies the potential performance gain or loss by the combined search/NSI routine. Since different NSIs vary in their distance estimate computations, the time ratio provides a more realistic characterization of search complexity (assuming sound implementations and equivalent processor speeds). Similar to the exploration ratio, values approaching zero indicate more efficient search.

3.2 Edge Weights

We first evaluated performance on networks with varying edge weight uniformity. We randomly applied weights to edges drawn from sets of possible weights with increasing cardinality. Given a single bin to draw from, the network is uniform, or unweighted. As the number of bins approaches or exceeds the number of edges, the number of distinct weights for the edges increases, resulting in a network that may have a unique weight for each edge.

We generated synthetic Forest Fire graphs [Leskovec et al. 2005], one of the best models of realistic social and Internet graphs to date, with 5,000 nodes, using a forward burning probability of 0.32 and a backward burning probability of 0.2. The generated networks typically had low density, with an average degree of three. The weights were randomly assigned to edges from a set of 1 to 10,000 bins. For each graph, we constructed the GNP and MATRIX FACTORIZATION NSIs with a logarithmic number of landmarks, DTZ with 10 zones and 10 dimensions, and SPT with three trees. Then, for 1,000 random pairs of nodes, each NSI was used to find approximate lowest-cost paths, and P, E, and T were averaged over 100 trials. Figure 2 shows the results (with 95% confidence intervals) of using the SPT, GNP, MF, and DTZ NSIs with A* search.

The results of this experiment suggest that each NSI can discover low-cost paths for arbitrarily weighted networks, with DTZ, MF, and SPT obtaining overall errors of less than 2%. GNP has slightly worse performance (although the error remains less than 10%), but exhibits an interesting phenomenon for graphs with a moderate number of distinct edge weights. For GNP and DTZ, the distance estimates deteriorate in quality with increasing granularity. As a result, the path ratio decreases, but the exploration ratio increases as the searches degenerate into A* search with a random heuristic function. MATRIX FACTORIZATION consistently produces an exploration

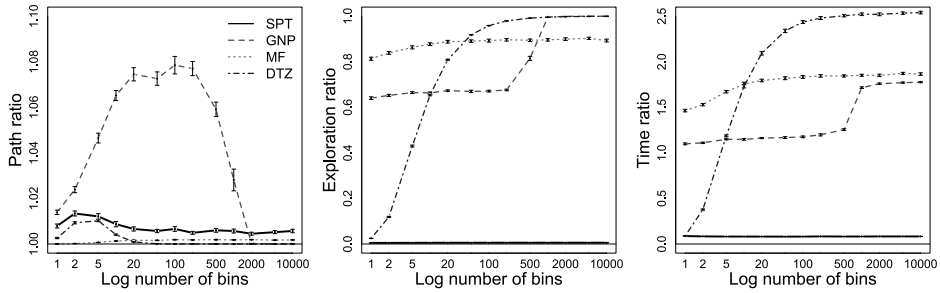


Fig. 2. NSI comparison on Forest Fire networks of 5,000 nodes. The cardinality of the set of possible edge weights varies from 1 to 10,000, essentially transforming the edge weights from uniform to distinctly random. Although all NSIs find highly accurate approximations to low-cost paths, only the SPT NSI does so efficiently. All other NSIs degrade with increasingly granular weighting schemes.

ratio of approximately 0.8, suggesting that its distance estimates correlate with actual distance. However, although each NSI finds highly accurate approximations to low-cost paths, only the SPT NSI does so efficiently. All other NSIs have unacceptable exploration and time ratios when weights are nonuniform. SPT finds paths of similar quality while exploring, on average, two to three orders of magnitude fewer nodes.

3.3 Network Density and Structure

Orthogonal to edge weights, a network can be characterized by its density. Network density is a measure of the ratio of links to nodes—the average degree of the network. We ran experiments on Forest Fire networks with low, medium, and high densities (0.32, 0.41, and 0.46 forward burning probabilities, respectively) yielding average degrees of three, five, and eight, and constructed NSIs on randomly weighted networks ranging from 1,000 to 10,000 nodes. The resulting performance measures for the SPT, GNP, MF, and DTZ NSIs exhibit little variation. The path ratio of SPT is lower than all other NSIs and is significantly better for high-density graphs. Each NSI performs slightly better with increasing density, an expected phenomenon since search becomes easier as nodes become more connected. In addition, SPT, with only three trees, finds paths in a small fraction of the time required by uniform-cost search. The other NSIs explore more than half the nodes required by uniform-cost search and are also significantly slower.

We also conducted experiments that test another aspect of NSI generalization—the structure of the network. Although Forest Fire networks have been shown to emulate desirable real-world network properties (e.g., power-law degree distribution, shrinking diameter), it is important to investigate the range of applicability of the NSIs. As such, we include results of NSI performance on lattice graphs [Watts and Strogatz 1998], slightly rewired with a probability of 0.01, and random networks [Erdos and Renyi 1959], even though these are more extreme forms of graph structure. All other experimental details are identical to those in the previous section. Figure 3 depicts the path, exploration, and time ratios of the four NSIs on the three classes of network structure, as the size of the graph increases.

SPT does not discover the lowest-cost paths on lattice and random graphs, but its performance remains acceptable. The rate at which SPT discovers paths is still orders of magnitude faster than for the other NSIs and uniform-cost search, and efficiency that does not compromise accuracy is our primary objective. However, with three trees, the SPT NSI introduces approximately 25–30% error on the length of the paths. Fortunately, lattice and random network structures do not occur regularly in domains targeted by social network analysis. These types of graphs exhibit the extremes of

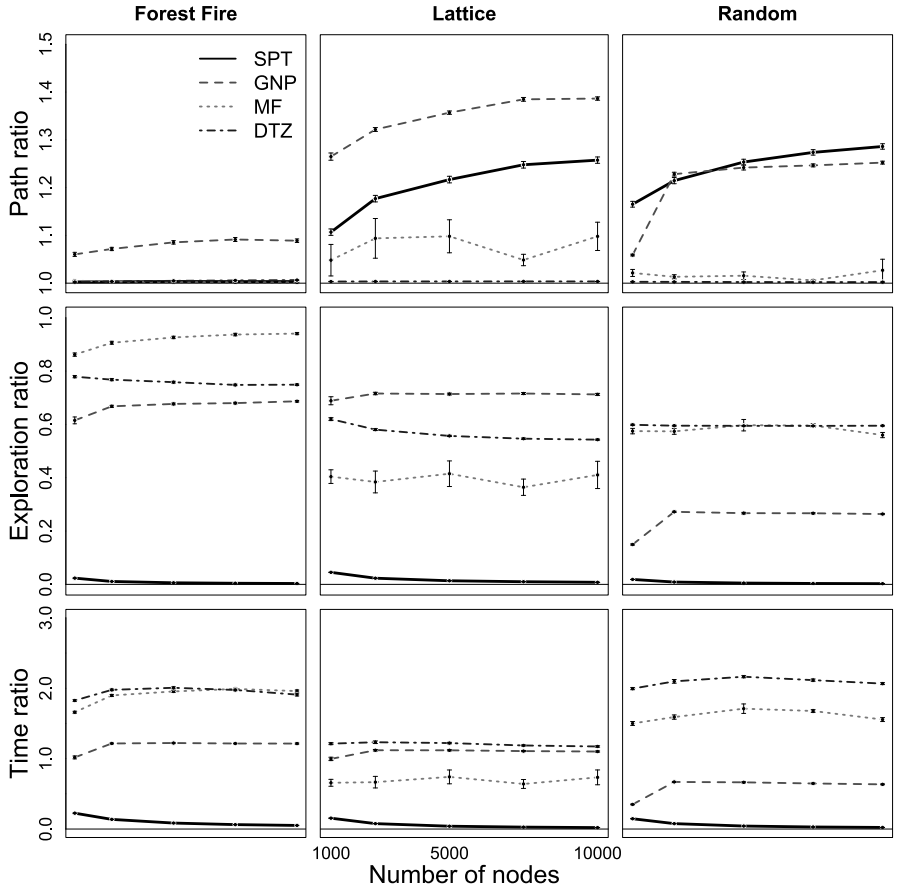


Fig. 3. Comparison of the SPT, GNP, MF, and DTZ NSIs on Forest Fire, lattice, and random networks. The SPT NSI offers acceptable performance in terms of path ratio, while exploring vastly fewer nodes.

structural properties, while the Forest Fire model generates graphs having realistic, small-world properties. Furthermore, we can improve the performance of the SPT NSI by increasing the number of shortest-path trees (see Section 3.5). The results on network structure and density demonstrate that the SPT NSI exhibits better general performance than other NSIs.

3.4 Distance Estimation

In order to provide evidence for how each NSI performs, we investigated the suitability of their distance estimates. For an NSI to quickly find good paths, its estimate of distance must have low error. Since we use A* search with the distance estimates as the heuristic function, the estimates must be accurate, not merely correlated with actual distance. Table II shows the mean squared error of the estimated distance with respect to the actual distances. The presented figures are an average of 30 mean squared errors calculated on Forest Fire networks with 10,000 nodes and 1,000 random pairs of nodes. The NSIs were constructed in the same manner as in previous sections. The superior accuracy of distance estimates for the SPT NSI is the primary reason it performs better than the other NSIs.

Table II. Mean Squared Error of Actual Distances Against NSI Distance Estimates for the SPT, MF, GNP, and DTZ NSIs

NSI	SPT	MF	GNP	DTZ
MSE	2.11	51.10	198.20	2482.73

The following two theorems provide guarantees on the lower and upper bounds of the distance estimates for the SPT NSI. The lower bound holds for arbitrary networks while the upper bound holds for arbitrary finite metric spaces.

THEOREM 3.1. *For arbitrary undirected graphs G, the distance estimate provided by the* SHORTEST-PATH TREE NSI *with any number of roots has a lower bound of the true lowest-cost distance,* $d(u, v) \leq d_{SPT}(u, v)$.

PROOF. Let $G = (V, E, w)$ be any undirected graph with an arbitrary edge weight function w. Let T_r be the shortest-path tree constructed by running Dijkstra's algorithm from an arbitrarily selected root node $r \in V$. By definition, the lowest-cost paths from r to all other nodes are included in T_r, so $\forall u \in V d_{T_r}(u, r) = d(u, r)$. Let $u, v \in V$ be two arbitrary nodes. Then, $d_{T_r}(u, r) + d_{T_r}(v, r) \geq d(u, v)$. If r is on the lowest-cost path from u to v, then $r \in \pi^*(u, v)$ and the equality holds; otherwise, $\pi^*(u, r) \cup \pi^*(v, r)$ follows a detour from the optimal path.

Let the SHORTEST-PATH TREE NSI consist of a single tree rooted at r. Then, $d_{T_r}(u, r) + d_{T_r}(v, r) \geq d_{SPT}(u, v) = d_{T_r}(u, r) + d_{T_r}(v, r) - 2d_{T_r}(i, r)$, where $i = u \cap v$, the lowest common ancestor of u and v in T_r. But, the distance between u and v computed via shortest-path tree T_r is equal to the distance from u to v along tree T_i, the shortest-path tree rooted at the lowest common ancestor. So, $d_{T_r}(u, r) + d_{T_r}(v, r) - 2d_{T_r}(i, r) = d_{T_i}(u, i) + d_{T_i}(v, i) \geq d(u, v)$. An SPT NSI with $l > 1$ trees has the same result since the minimum distance over all trees will still be bounded below by $d(u, v)$. Therefore, for an arbitrary graph and any two nodes, we have $d_{SPT}(u, v) \geq d(u, v)$. □

Intuitively, the SPT NSI provides distance estimates derived only from actual paths in the graph, so it will never underestimate the true lowest-cost distance between any pair of nodes. Next, we prove that the SPT NSI has stretch 3 distance estimates (i.e., has an upper bound of three times the actual distance) for all but an ϵ-fraction of the node pairs with high probability and using only a constant number of trees. The following theorem is adapted from a result in Kleinberg et al. [2009].

THEOREM 3.2. *For any finite metric space, a* SHORTEST-PATH TREE NSI *with a constant number of randomly selected roots has stretch 3,* $d_{SPT}(u, v) \leq 3d(u, v)$*, for all but an* ϵ*-fraction of the node pairs with probability of at least* $1 - \gamma$.

PROOF. Let $G = (V, E, w)$ be a finite metric space with $|V| = n$, and let $\epsilon, \gamma \in (0, 1)$. Let B_u be the smallest ball of nodes around u such that $|B_u| \geq \epsilon n/2$. So, $\forall w \in B_u \quad d(w, u) \leq r$, for some distance r. For every node u, at least one node in B_u will be selected as a root with probability $1 - \epsilon\gamma/2$ if an SPT NSI randomly selects a constant number of root nodes. The number of trees can be derived by solving for l in the equation $1 - (1 - \epsilon/2)^l = 1 - \epsilon\gamma/2$, yielding $l = \dfrac{\log(\epsilon\gamma/2)}{\log(1 - \epsilon/2)}$ root nodes.

So, with probability $1 - \epsilon\gamma/2$, a node $b \in B_u$ will be selected as a root. Then, let $v \in V$ be one of the at most $n - \epsilon n/2$ nodes that are either outside B_u or on its

periphery. The distance estimate $d_{\text{SPT}}(u, v)$ provided by the SPT NSI is at most the distance estimate provided by the single shortest-path tree rooted at b, $d_{\text{SPT}}(u, v) \leq d_{T_b}(u, v)$, and $d_{T_b}(u, v) = d_{T_b}(u, b) + d_{T_b}(v, b) - 2d_{T_b}(u \cap v, b)$, by definition of the SPT NSI distance function. Then, we have

$$\begin{aligned}
d_{\text{SPT}}(u, v) &\leq d_{T_b}(u, v) \\
&= d_{T_b}(u, b) + d_{T_b}(v, b) - 2d_{T_b}(u \cap v, b) \\
&\leq d_{T_b}(u, b) + d_{T_b}(v, b) && \text{removal of last quantity} \\
&= d(u, b) + d(v, b) && \text{definition of shortest-path trees} \\
&\leq d(u, v) + d(v, b) && \text{by radius of } B_u \\
&\leq d(u, v) + d(u, v) + d(u, b) && \text{by the triangle inequality} \\
&\leq d(u, v) + d(u, v) + d(u, v) && \text{by radius of } B_u \\
&= 3d(u, v).
\end{aligned}$$

So, for all but at most $\epsilon n/2$ node pairs involving u, we have $d_{\text{SPT}}(u, v) \leq 3d(u, v)$.

Next, let $I_u = \begin{cases} 1 & \text{if root} \in B_u \\ 0 & \text{otherwise} \end{cases}$ be an indicator variable of nodes for which the desired bound holds. Then, $p(I_u = 1) \leq \epsilon\gamma/2$. If $N = \sum_{u \in V} I_u$, then $E[N] \leq \epsilon\gamma n/2$ and, by the Markov inequality, $p(N > \epsilon n/2) < \gamma$. Then, the maximum number of node pairs not guaranteed to be bounded by stretch 3 is $\leq \sum_{u \in V} \begin{cases} \epsilon n/2 & \text{if } I_u = 0 \\ n & \text{if } I_u = 1 \end{cases} \leq \left(n - \frac{\epsilon n}{2}\right) \cdot \frac{\epsilon n}{2} + \frac{\epsilon n}{2} \cdot n = \frac{\epsilon n}{2}\left(2n - \frac{\epsilon n}{2}\right) \leq \epsilon n^2$. This proves that the SPT NSI distance estimate has an upper bound of three times the actual distance for all but an ϵ-fraction of node pairs using a constant number of trees with probability $1 - \gamma$. □

Theorem 3.2 holds for arbitrary finite metric spaces. As a result, the theorem applies to all unweighted graphs, but an arbitrarily weighted network is not necessarily a metric space since the distance function may violate the triangle inequality. Also, the theorem suggests that in order to guarantee stretch 3, the SPT NSI may need a large, constant number of trees. For example, with $\epsilon, \gamma = 0.1$, 103 trees are required. In practice, however, the SPT NSI has extremely accurate distance estimates with only a small number of trees (all previous experiments were performed with only three trees). This is because the distance function exploits the lowest common ancestor for individual pairs of nodes within each shortest-path tree. The lowest common ancestor is effectively a root of a subtree for a local region of the graph, which provides tighter distance estimates.

In order to determine the actual benefit provided by this extra computation, we directly compare a landmarks NSI that uses the same nodes for landmarks as were used for roots for the trees in the SPT NSI. The landmarks distance function provided by Potamias et al. [2009] is identical to the first component of the SPT NSI distance function. The large reduction in error of distance estimates over pure landmarks is shown in Figure 4 on unweighted and weighted Forest Fire networks, as well as on lattice and random networks. This reduction in error can be completely attributed to the additional component of the distance function that exploits the lowest common ancestor within shortest-path trees. In the next section, we investigate the accuracy of the distance estimates across network structures for varying numbers of trees.

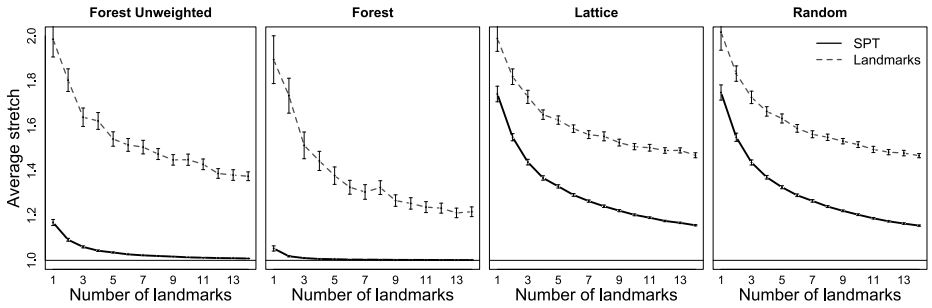


Fig. 4. Average stretch ratios of a landmarks NSI and the SPT NSI on unweighted and weighted Forest Fire networks and weighted lattice and random networks with varying numbers of landmarks.

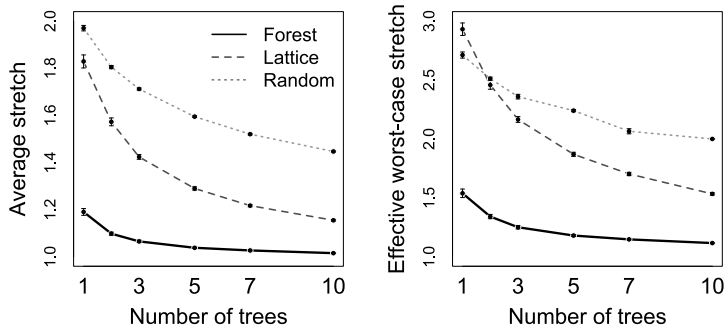


Fig. 5. Average and effective worst-case (95% threshold) stretch ratios on Forest Fire, lattice, and random networks with varying number of shortest-path trees composing the NSI. The NSI provides near-optimal distance estimates for Forest Fire networks and low stretch for lattice and random networks.

3.5 Number of Shortest-Path Trees

The experiments in the previous sections indicate that the SPT NSI offers performance that generalizes across arbitrary weighting schemes on networks of various structures and densities. The results suggest that only a few trees are required for accurate distance estimation and low-cost path finding. However, it is important to investigate the single free parameter of this NSI; that is, how is performance affected by the choice of the number of trees? In this section, we present results on SPT NSI distance estimation and searching performance on three graph structures as the number of trees is varied.

While Theorem 3.2 provided an upper bound for the distance estimates, it is difficult to parameterize tight theoretical bounds for the worst-case and average-case stretch error in terms of the number of shortest-path trees in the NSI. This is compounded when considering arbitrary network structure, let alone edge weights. Therefore, we provide an empirical evaluation for the networks explored thus far (Forest Fire, lattice, and random graph structures). Figure 5 shows the average stretch and effective worst-case stretch, the 95th percentile stretch, for the SPT NSI on unweighted networks of 10,000 nodes as a function of the number of shortest-path trees. We choose 1,000 random pairs of nodes to estimate these statistics and provide 95% confidence intervals over 100 trials for each network and NSI pair. The NSI provides near-optimal distance estimates for Forest Fire networks and low stretch error for lattice and random networks. As expected, small increases in the number of shortest-path trees can

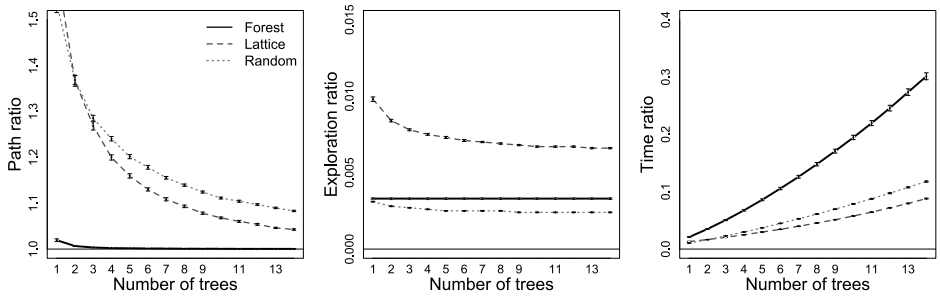


Fig. 6. Performance characterization on Forest Fire, lattice, and random networks with varying number of shortest-path trees used to create the NSI. In lattice and random networks, the errors in path costs decrease substantially with increasing numbers of trees. Forest Fire networks require few trees to discover low-cost paths.

significantly reduce the error of approximation, and clearly, the error monotonically decreases to 1.0 (optimal) as the number of trees approaches the size of the graph. For three trees (as in the preceding experiments), the NSI provides average stretches of 1.07, 1.42, and 1.71, and effective worst-case stretches of 1.25, 2.17, and 2.36 for Forest Fire, lattice, and random networks, respectively. We repeated these experiments for weighted networks with very similar results.

Figure 6 shows the path, exploration, and time ratios for the SPT NSI on Forest Fire, lattice, and random graphs of 10,000 nodes as a function of the number of trees comprising the NSI. The number of trees varies from a single tree to $\log n$ trees, where n is the number of nodes. As expected, adding trees can never decrease path performance. For Forest Fire graphs, however, it is clear that only a small number of trees is necessary to attain high performance. In fact, it is more beneficial to choose a small number since the cost of the distance function increases linearly with the number of trees. For the more extreme lattice and random network structures, a balance must be found between path approximation and cost. For error-tolerant applications, we need only construct a few trees to gain efficiency; for cost-tolerant applications, more trees will increase accuracy. Fortunately, most real-world graphs exhibit small-world properties, making this trade-off less critical.

It is also possible to automatically determine the appropriate number of trees, l, given a convergence threshold on distance estimates. We can employ a strategy, similar to the approach used in progressive sampling [Provost et al. 1999], in which we progressively double the size of the NSI. This procedure begins with an SPT NSI constructed with a single tree, computes its distance estimates on a sample of node pairs, and progressively increases the size of the NSI until the estimates have converged. Without any a priori knowledge about the network, this technique will construct an SPT NSI, determining the appropriate setting for l using an asymptotically optimal schedule [Korf 1985]. We present the pseudocode in Algorithm 1.

4. APPLICATIONS

The purpose of a network structure index is to provide useful estimates of distances to enable efficient navigation. The primary benefit is that, for the one-time cost and additional storage requirement of constructing such an index, a variety of applications become tractable. In this section, we present results on desirable network applications, including navigation, diameter approximation, centrality calculation, and clustering. In addition to experiments on synthetic networks, we provide example applications on three real data sets.

ALGORITHM 1: ProgressiveSPTNSI(G, δ)

```
// Progressively build an SPT NSI for graph G by doubling the number of trees
// until the change in estimates between iterations is less than the given
// threshold δ
l ← 1
N_r ← RandomNodePair(G) × 1000                          // sample pairs of nodes
D' ← BuildSPT(G, l)                          // construct initial NSI with single tree
d ← Distance(D, N_r)                              // get distance estimates for sample
δ' ← ∞
while δ' > δ do
    D ← D'
    D' ← AddRoots(D, G, l)                                // add additional trees to NSI
    d' ← Distance(D', N_r)                               // recompute distance estimates
    δ' ← ( Σ_i (d_i − d'_i) / d'_i ) / |d| − 1                    // compute average change
    l ← 2 × l                                    // progressively double size of SPT NSI
return(D)
```

DBLP[1]. The Digital Bibliography & Library Project offers data on authors and publications in various computer science fields. We use a subset of this data set, namely the coauthorship network as of 2006. This network consists of 343,458 author objects connected by 1,145,564 coauthorship links. It is a sparse graph, with an average degree of roughly 3.33. To form a weighted network, we add weights as the inverse normalized coauthorship count between two authors.

IMDb[2]. The Internet Movie Database is an authoritative source for data relating to the movie industry. We use a subset of the actor costarring network, in which we only consider movies released between 1970 and 2000 and actors with a minimum of 20 appearances. This results in 11,132 actor objects connected by 1,229,550 costarring links. This is an extremely dense graph, with an average degree of approximately 110. We apply edge weights as the inverse normalized costar count between two actors.

FINRA[3]. The Financial Industry Regulatory Authority is the primary private-sector regulator of the securities industry in the United States. FINRA is responsible for overseeing the activities of all securities firms, their branch offices, and the individuals (reps) that work for them. The complexity of this industry was recently captured in a relational data set [Fast et al. 2007], and we use a subset that captures the employment movement of reps among branches [Friedland and Jensen 2007]. Specifically, 428,356 branch objects are connected via 1,394,832 links. This is the largest connected component of all possibly connected branches. Two branches are connected if a rep transferred at any time from one branch to the other. This is another sparse graph, with an average degree of 3.26. Edge weights are assigned as the inverse of the average of the two proportions of the number of transferring reps to the total number of reps working at the branches.

[1]http://dblp.uni-trier.de/
[2]www.imdb.com
[3]www.finra.org

Table III. Path, Exploration, and Time Ratios of the SPT NSI on Three Real Networks

Data set	Path ratio	Exploration ratio	Time ratio
DBLP	1.0218	0.0001	0.0084
IMDb	1.0978	0.0011	0.0876
FINRA	1.0738	<0.0001	0.0300

4.1 Navigation

Our first task is navigation in a network. Table III reports the path, exploration, and time ratios for the SHORTEST-PATH TREE NSI on the DBLP, IMDb, and FINRA data sets, each averaged over 10 trials of a newly constructed NSI and 1,000 random pairs of nodes. The NSI uses 10 trees because some of these networks are substantially larger than those in the synthetic networks from Section 3. Note that the time required to search is a function of the size and density of the network and the number of trees composing the NSI. For this particular experiment, searching 1,000 pairs of nodes with an NSI consisting of 10 trees required only several minutes.

These results suggest that the SPT NSI can efficiently find low-cost paths in real networks. Additionally, based on the analysis of path ratio and the number of shortest-path trees (see Section 3.5), we can make conjectures about network structure. First, the number of nodes in the IMDb network is roughly equivalent to that used in the synthetic experiments, and the path ratio is roughly 1.1. Therefore, it is probable that IMDb has a random network structure, which is plausible given that the core is known to consist of a close-knit web of relationships. Similarly, the structure of DBLP may be more akin to the Forest Fire structure, since its path ratio is much lower, despite comprising approximately 30 times more nodes. This is also plausible as the growth of coauthorship networks has been shown to follow preferential attachment [Pollner et al. 2006]. Based on its size and path ratio, the FINRA structure may lie between a Forest Fire and a random network.

4.2 Graph Diameter

We can approximate the diameter (maximum lowest-cost distance) of any graph using a network structure index. Furthermore, the effective diameter, or 90^{th} percentile distance—a more robust statistic [Tauro et al. 2001]—can easily be determined by searching over a random sample of node pairs. In this section, we only consider the unweighted versions of our data sets since the diameter with weighted edges is less meaningful in the current context. For networks with edges corresponding to distance (e.g., latency or transit measurements), as opposed to strength of a relationship, the analysis of graph diameter is more informative.

By finding the paths over a given sample of node pairs, we can estimate any distance percentile. We present the results for our three data sets using 10,000 sampled node pairs and computing the 50th, 90th, and 100th percentiles, or the median path length, effective diameter, and actual diameter. We discover the following.

— IMDb has a median distance of 3, an effective diameter of 4, and a maximum distance of 5.
— DBLP has a median distance of 7, an effective diameter of 9, and a maximum distance of 17.
— FINRA has a median distance of 5, an effective diameter of 6, and a maximum distance of 11.

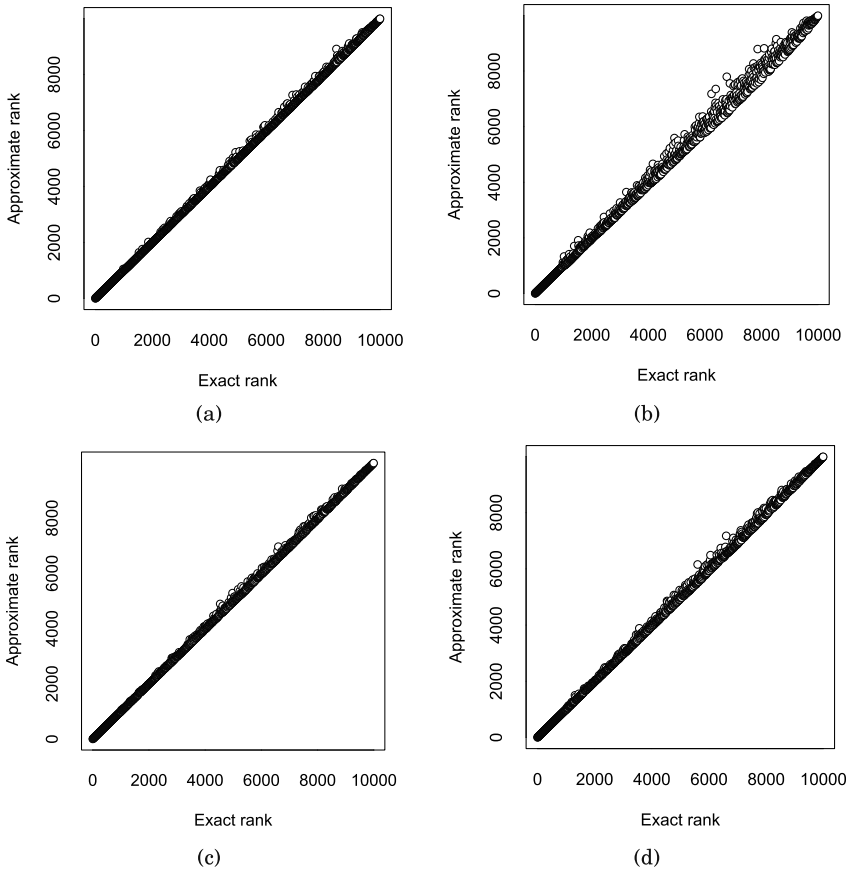


Fig. 7. Ranking comparisons for closeness centrality approximation on synthetic Forest Fire networks with 10,000 nodes using an SPT NSI with three trees and 10,000 sampled pairs. (a) unweighted, searching; (b) unweighted, distance estimates; (c) weighted, searching; (d) weighted, distance estimates.

4.3 Closeness Centrality

A standard analysis of networks involves identifying nodes with the greatest centrality. There are various notions of centrality, and here we demonstrate the effectiveness of using a network structure index for approximating closeness centrality. Closeness centrality is a simple measure of the proximity of a node to all other nodes in a network [Freeman 1979].

$$C(u) = \sum_{v \in V} d(u, v).$$

The closeness centrality of any given node is the sum of the distances to all other nodes in the network. Thus, to calculate closeness centrality exactly, the cost of all pairwise distances must be determined. To estimate closeness centrality, we can simply calculate the average distance to a sample of nodes. Typical analysis is primarily concerned with identifying a set of nodes that have the greatest centrality, and an NSI can be used to efficiently approximate centrality rankings [Rattigan et al. 2006].

Figures 7(a) and 7(b) depict scatterplots of exact versus approximate closeness rankings obtained using an SPT NSI containing three trees on an unweighted synthetic

Table IV. The Top 10 Actors with the Greatest Closeness Centrality in IMDb for Unweighted (left) and Weighted (right) Versions of the Network

1. Klaus Maria Brandauer	1. Herschel Savage
2. Griffin Dunne	2. Vernon Dobtcheff
3. Dominique Pinon	3. Joey Silvera
4. Jorge Cervera Jr.	4. Peter North
5. Louise Sorel	5. Tom Byron
6. Urbano Barberini	6. Jon Dough
7. Walter Koenig	7. Ron Jeremy
8. Wadeck Stanczack	8. Jonathan Morgan
9. Martin Sheen	9. T. T. Boy
10. Mario Adorf	10. Randy West

Forest Fire network with 10,000 nodes and a sample of 10 for each node. Note that for this experiment, even the so-called exact centrality ranks are actually estimates obtained by exact determination of the distance of any given node to the same sample of 10 nodes. Since the SPT NSI provides accurate distance estimates, a more efficient alternative to searching is to use the distance estimates directly. As is evident from these results, using paths or distance estimates based on the NSI provides highly accurate approximations of centrality. The quality of a ranking can be evaluated using Spearman's ρ and Kendall's τ statistics. When searching with the NSI, we achieve values of 0.9999 and 0.9944, while the distance estimates achieve 0.9985 and 0.9706. A slightly more informative measure is the precision of the top n nodes in the ranking; our ranking produces precision values of 1.0 and 0.999 for the top 100 and 1000 nodes when approximating closeness with searching, and 1.0 and 0.987 using the distance estimates. This clearly demonstrates the accuracy of the SPT NSI as an estimator of closeness centrality. Additionally, even though the distance estimates are slightly less accurate than those obtained by searching, the NSI provides estimates in constant time, which in practice, allows for significantly more pairs to be sampled.

We repeated the experiment on weighted Forest Fire networks (see Figures 7(c) and 7(d)). The approximation is roughly equivalent to its unweighted counterpart. The values for ρ, τ, precision at 100, and precision at 1000 are 0.9998, 0.9925, 1.0, 0.992 for searching and 0.9996, 0.9869, 1.0, 0.99 using the distance estimates.

We also identified the 100 most central actors in IMDb and authors in DBLP, using the distance estimates provided by a SHORTEST-PATH TREE NSI with 10 trees. Treating a network as having weighted edges can greatly affect the resulting analysis. The 100 most central actors in the unweighted and weighted versions of IMDb have a single actor in common (Vernon Dobtcheff), and the 100 most central authors in DBLP have a 4% overlap. Tables IV and V present the top 10 lists for the unweighted and weighted versions of the IMDb and DBLP networks using 1,000 random distance estimates per node. Note that for IMDb, there is a substantial portion of the network related to adult film. According to IMDb, adult film stars appear in many more films than regular actors and frequently work with many of their colleagues. Consequently, for the weighted IMDb network, adult film stars have a disproportionate number of high closeness actors, with 9 out of the top 10 actors belonging to that community.

4.4 Betweenness Centrality

Betweenness centrality is a gauge of how critical a given node is to the structure of the network [Freeman 1979]. Nodes with high betweenness centrality often act as a

Table V. The Top 10 Authors with the Greatest Closeness Centrality in DBLP for Unweighted (left) and Weighted (right) Versions of the Network

1. Toby J. Teorey	1. Ming Li
2. Pierre Fraigniaud	2. Paul M. B. Vitanyi
3. Arvind Krishnamurthy	3. Tao Jiang
4. Tharam S. Dillon	4. Eitan M. Gurari
5. Victor C. M. Leung	5. Zhe Dang
6. Volker Linnemann	6. Wei Wang
7. W. Bruce Croft	7. Jeffrey D. Ullman
8. Ahmed Amine Jerraya	8. Jennifer Widom
9. Shamkant B. Navathe	9. John Tromp
10. Yossi Matias	10. Richard Hull

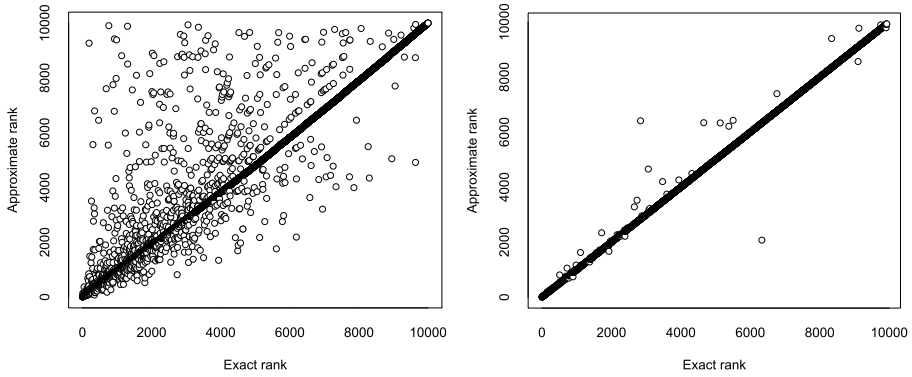


Fig. 8. Ranking comparisons for betweenness centrality approximation on unweighted (left) and weighted (right) synthetic Forest Fire networks with 10,000 nodes using an SPT NSI with three trees and 50,000 sampled pairs.

bridge between communities, or clusters, of a network. The betweenness centrality of a node, u, is the proportion of all shortest, or lowest-cost, paths that the node lies upon.

$$B(u) = \sum_{v,w \in V} \frac{g_u(v, w)}{g(v, w)}, u \neq v \neq w,$$

where $g(v, w)$ is the number of lowest-cost (geodesic) paths connecting nodes v and w, and $g_u(v, w)$ is the number of these paths that pass through node u. In order to obtain an exact measure of betweenness, all lowest-cost paths between all pairs of vertices must be enumerated. This is often intractable in practice, and the best known algorithm requires $O(nm)$ for unweighted networks and $O(nm + n^2 \log n)$ for weighted networks [Brandes 2001]. There has also been considerable research on approximating betweenness centrality [Brandes and Pich 2007; Geisberger et al. 2008], and since NSIs can rapidly produce low-cost paths, we demonstrate how the SPT NSI can be used to identify high-betweenness nodes in unweighted and weighted contexts.

Figure 8 depicts two scatterplots of exact versus approximate betweenness rankings obtained with an SPT NSI with three trees on an unweighted and weighted synthetic Forest Fire network with 10,000 nodes and 50,000 sampled pairs of nodes. Again, the "exact" centrality ranks are actually estimates obtained by sampling the same 50,000 node pairs, determining the shortest path between them, and then incrementing a count for all nodes on that path. These results show that using an NSI to approximate

Table VI. The Top 10 Actors with the Greatest Betweenness Centrality in IMDb for Unweighted (left) and Weighted (right) Versions of the Network

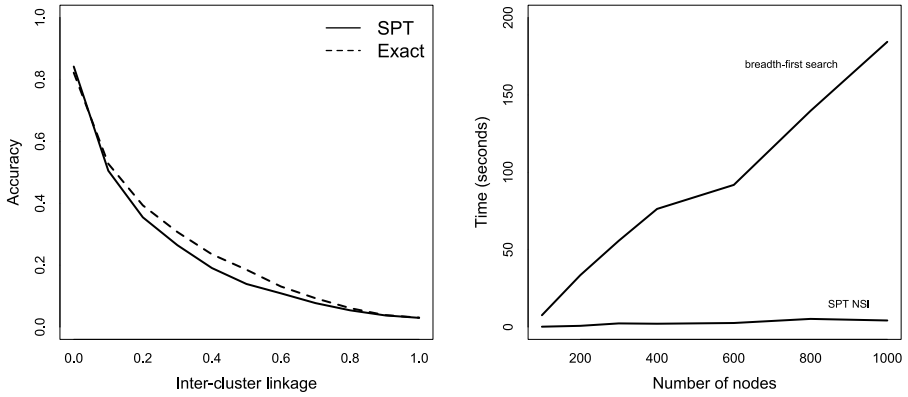
1. Martin Sheen	1. Frank Welker
2. Ron Jeremy	2. Tom Byron
3. Joey Silvera	3. Vernon Dobtcheff
4. Max Von Sydow	4. Ron Jeremy
5. Andréa Ferréol	5. Brion James
6. Elliot Gould	6. Max Von Sydow
7. Mario Adorf	7. Gérard Depardieu
8. Christopher Lee	8. John Gielgud
9. Vernon Dobtcheff	9. Saveli Kramarov
10. Frank Welker	10. Leonid Kuravlyov

Table VII. The Top 10 Authors with the Greatest Betweenness Centrality in DBLP for Unweighted (left) and Weighted (right) Versions of the Network

1. Tharam S. Dillon	1. Ming Li
2. Shamkant B. Navathe	2. Tao Jiang
3. Matthias Jarke	3. Philip S. Yu
4. W. Bruce Croft	4. Oscar H. Ibarra
5. Joachim W. Schmidt	5. Jiawei Han
6. Won Kim	6. Stefano Ceri
7. Howard Jay Siegel	7. Jianwen Su
8. Umeshwar Dayal	8. Jennifer Widom
9. Wolfgang Rosenstiel	9. Jeffrey D. Ullman
10. Wayne Wolf	10. Masahiro Fujita

betweenness centrality is highly effective, providing accurate approximations with only a fraction of the cost (since NSI-guided search is much faster than exact search). The ranking achieves values for Spearman's ρ and Kendall's τ of 0.9633 and 0.9266 for the unweighted network and 0.9995 and 0.9975 for the weighted network. The precision results at the top 100 and 1000 nodes are 0.86 and 0.907 for the unweighted network and 0.98 and 0.993 for the weighted network. The NSI performs better for weighted networks since there is typically a single lowest-cost path between an arbitrary pair of nodes, whereas the unweighted network may have multiple geodesics.

We also approximated the betweenness centrality for the real networks IMDb and DBLP, which would be computationally intensive to calculate with exact methods. The 100 most central actors in the unweighted and weighted versions of IMDb have a 26% overlap, and the 100 most central authors in the unweighted and weighted versions of DBLP have a 21% overlap. This suggests that the nodes occurring on short paths are less affected by edge weights than by the distances themselves since the nodes with high closeness centrality have small overlap. In Tables VI and VII, we list the top 10 actors and authors with the highest estimated betweenness centrality in the unweighted and weighted versions of both networks. These nodes were identified by constructing an SPT NSI with 10 trees on each network, using 100,000 searches over the network, and ranking the nodes by their appearances in these searches.



(a) (b)

Fig. 9. (a) Clustering accuracy of k-medoids using the SPT NSI and breadth-first search on synthetic networks with 1,000 nodes. The SPT NSI yields highly competitive clusters compared to the exact method. (b) The run time of the clustering algorithm using breadth-first search increases rapidly as the size of the graph increases; clustering with the NSI requires a fraction of the time.

4.5 Clustering

Discovering latent communities in networks is an important task in knowledge discovery, and simple algorithms for clustering graphs can be highly effective [Newman 2004b]. Rattigan et al. [2007] have shown that network structure indices can drastically reduce the complexity of exact clustering methods. In this section, we demonstrate how the SPT NSI can be used in conjunction with the graphical k-medoids algorithm (as described by Rattigan et al. [2007]). Through synthetic experiments, we show that using the SPT NSI to cluster networks offers comparable performance to the exact method while requiring only a fraction of the time. Figure 9 depicts the results of clustering synthetic networks generated and evaluated using the same techniques introduced in Section 2.1 in Rattigan et al. [2007]. As expected, the accuracy of clustering decreases as the difficulty of the clustering problem increases. This is primarily due to the reduced structural separation of clusters as the intercluster linkage increases. However, the notable result here is that the SPT NSI does not substantially reduce accuracy. Figure 9 also presents the runtime as a function of graph size for the k-medoids algorithm when using breadth-first search and the SPT NSI. This graph shows that clustering with breadth-first search can quickly become intractable as the size of the data increases. Using the network structure index alleviates this problem, requiring a fraction of the time for breadth-first search.

We also used the SPT NSI to cluster the weighted FINRA and IMDb networks. The FINRA network consists of over 400,000 branches connected by the employment transfers of representatives [Friedland and Jensen 2007]. We ran the k-medoids algorithm using the SPT NSI with three trees to discover 400 clusters. The algorithm converged quickly, requiring only three iterations. To evaluate the clustering, we used background knowledge of the branches, specifically geography (the state the branch is located in) and affiliation (the firm the branch is associated with). We ran a likelihood ratio test against the null hypothesis that the states and firms are drawn from a multinomial distribution, given the baseline rates for each state and firm. The test was highly significant, suggesting that the algorithm discovered clusters that were geographically constrained and tied to particular firm affiliations.

Finally, the IMDb data set can be clustered to discover groups of actors that are closely related based on common appearances in movies. We found that using an

SPT NSI with 10 trees successfully discovers communities that consist entirely of a single nationality (e.g., French, Russian, Asian, Italian, Scandinavian) or of a particular genre (e.g., professional wrestling, political figures appearing in documentaries, pornography). The clustering also found interesting groups of related actors, such as those focused around James Bond films and those tied to Harrison Ford in Star Wars and Indiana Jones movies. Again, the clustering algorithm converged after several iterations, which is made tractable by exploiting the network structure index.

5. RELATED WORK

Searching in small-world networks has been a topic of interest since the well-known study of Travers and Milgram [1969], in which participants in Omaha, Nebraska were asked to forward a letter to individuals in Boston, Massachusetts. The major finding of this experiment was that letters reaching the destination did so in relatively few transfers. Since then, researchers have attempted to expose the mechanisms and characteristics of networks under which search is feasible.

Distance estimation. Graph preprocessing to enable efficient navigation and distance estimation is the objective of this work and has been studied in several academic disciplines. For instance, the graph labeling problem has been studied extensively in the graph theory community. Here, the graph is preprocessed, and each node is annotated with a label. These labels are then used for estimating the distance between nodes [Gavoille et al. 2001]. Several algorithms with varying time and space complexity guarantees exist for exact and approximate distance calculations [Cohen and Zwick 2001; Dor et al. 1996]. Zwick [2001] presents an excellent survey of several techniques and their associated applications. In more recent work, Thorup and Zwick [2005] present a "distance oracle" approach to the distance estimation task. For finite metric spaces, the algorithm is guaranteed to exhibit constant query time, $O(mn^{\frac{1}{2}})$ processing time, and $O(n^{\frac{3}{2}})$ space to guarantee stretch 3 for all pairs of nodes. As stated in Theorem 3.2, the SPT NSI is more efficient in terms of time and space complexity ($O(m \log n)$ and $O(n)$, respectively). However, as described in Section 3.4, our approach guarantees stretch 3 for all but an ϵ-fraction of the node pairs.

In the networking community, the well-known problem of estimating the latency between nodes on the Internet has attracted many researchers and produced various approaches. The most common technique is to embed the network into a low-dimensional space that approximates the delay space. The first such approach, GNP [Ng and Zhang 2002], embeds the distances among a small set of landmarks into a Euclidean space using simplex optimization. Then, all other nodes are embedded within the same coordinate space by computing the distances to the landmarks. VIRTUAL LANDMARKS [Tang and Crovella 2003] uses a similar approach to GNP. The distances to landmarks are used as coordinates for a Lipschitz embedding, and then principal components analysis determines a lower-dimensional embedding. LIGHTHOUSE [Pias et al. 2003] extends the basic ideas of GNP but uses multiple coordinate systems in order to reduce communication bottlenecks for distributed systems. This approach has comparable performance to GNP, and it optimizes for a different task than our current objective of local social network analysis.

As opposed to embedding based on landmarks, several researchers investigated another class of techniques for network embeddings based on simulations. VIVALDI [Dabek et al. 2004] treats the network as a system of springs and attempts to find a stable relaxation of the analogous physical system. BIG-BANG SIMULATION [Shavitt and Tankel 2004] similarly models the set of nodes as particles influenced by a potential force field.

Strategies other than network embeddings have been proposed in the networking literature. One approach is based on rings of neighbors, as introduced by Slivkins [2007], and used in the practical, distributed system, MERIDIAN [Wong et al. 2005]. The results require that the network be a doubling metric, and it is unclear how widely applicable this technique is to arbitrary graphs and edge weightings. MATRIX FACTORIZATION [Mao and Saul 2004] was developed as a way to overcome violations of the triangle inequality and asymmetry observed in Internet routing times. We demonstrated in this article that the SPT NSI significantly outperforms MATRIX FACTORIZATION and network embeddings for arbitrary structure and edge weightings.

In the knowledge discovery literature, the work of Rattigan et al. [2006] proposed network structure indices to enable efficient network navigation as a tool to approximate network properties, such as centrality measures. The most accurate approach presented in that work was DISTANCE TO ZONE; however, its applicability is limited to unweighted networks and both the time and space complexity are prohibitive for practical usage.

A comprehensive theory of landmark, or beacon-based, methods and embeddings is presented by Kleinberg et al. [2009]. By definition, the triangulation for distance estimation used by beacon-based methods are greater than or equal to the distance estimates provided by the SPT NSI. Consequently, all theoretical guarantees on the upper bound of distance estimation also apply to the SPT NSI, but are necessarily looser upper bounds. We showed in Section 3.4 a widely applicable theoretical guarantee on the upper bound of the distance estimates, as well as empirical evidence that the SPT NSI is highly accurate.

While the theoretical results of Kleinberg et al. [2009] imply that random landmark selection can lead to provably accurate distance estimates, there has been some work showing that additional performance gains can be achieved in practice. Although identifying the optimal set of landmarks is $\mathcal{NP}$-hard, heuristics based on centrality and network structure [Potamias et al. 2009] or clustering [Tang and Crovella 2004] can be used to outperform random selection. These methods have the potential to improve the performance of the SPT NSI by identifying which nodes to select as roots of the shortest-path trees.

Searching with intrinsic properties. Apart from preprocessing techniques that facilitate search, there is also a large body of research that focuses on how navigation can be assisted by the natural properties of small-world networks. Adamic et al. [2001] examine algorithmic methods that utilize network hubs (nodes with high degree) for navigation. Kurumida et al. [2006] present a general framework for characterizing structure-based search strategies, as does Arcaute et al. [2007]. Since the solution space of several classic problems in computer science can be characterized as small-world graphs, structure-based methods can be utilized for efficiently finding solutions to problems such as graph coloring and satisfiability [Roli 2005; Walsh 1999].

Besides a unique network structure, many small-world networks are characterized by homophily—the tendency of neighboring nodes to have similar attribute values. Like degree structure, this property can be exploited for navigation and search. Kleinberg [2000a, 2000b] investigates the searchability of networks given an underlying model, as do Watts et al. [2002] and Kumar et al. [2006]. Şimşek and Jensen [2008] combine the structure-based and attribute-based approaches in a probabilistic framework. The approaches that use the inherent structural and attributional properties of networks are solely concerned with efficient navigation between pairs of nodes. Although efficiency is a goal of the current work, we maintain the additional objective that the identified paths approximate the lowest-cost paths.

Related tasks. There has been a considerable amount of recent research on discovering meaningful connections between nodes, other than through the lowest-cost paths. The work by Faloutsos et al. [2004], and more recently, Koren et al. [2007], attempts to find subgraphs involving arbitrary nodes that are representative of the important relationships between them. These paths, discovered through random walks, do not necessarily include the lowest-cost paths, nor are they approximations to them. Connection subgraphs and proximity graphs have been demonstrated to be of relevance for various applications and domains, but the objectives of the present work and this area of research do not coincide.

A related task is that of finding the K shortest paths between pairs of nodes. This is strictly a more challenging task than identifying a single approximate low-cost path between a pair of nodes. An efficient algorithm that solves this problem, for a single pair of nodes, is given by Hadjiconstantinou and Christofides [1999] in $O(Kn^2)$ time. This approach would not be appropriate for the network analytic tasks described in this article, which require efficient estimation of distances.

6. CONCLUSIONS AND FUTURE WORK

We have developed an index of network structure, based on the shortest-path tree, which offers better overall performance than other known indices. The SHORTEST-PATH TREE NSI has been shown to provide more accurate distance estimates on a range of arbitrarily weighted networks with different structures and densities. The complexity of the index is dominated by several applications of Dijkstra's single-source shortest-path algorithm and is linear in its space complexity. This makes the SPT NSI practical for deployment on real data sets and their applications.

The research on network structure indices has not yet been exhausted, and several open questions remain. In the work presented here, we chose the root nodes of the shortest-path trees randomly. Since the number of landmark nodes is small, it is possible that strategic approaches, such as clustering or using centrality measures, that maximize coverage might improve performance [Potamias et al. 2009; Tang and Crovella 2004]. However, for large networks this is computationally expensive, and random selection has also been demonstrated to be quite effective. Finally, we would like to develop a network structure index capable of handling directed graphs. Arbitrary undirected networks are the primary focus of this work, but in order to completely generalize applicability, we should allow for edge orientation.

ACKNOWLEDGMENTS

The authors wish to thank Cindy Loiselle for her helpful comments and Kevin Fall for recommending stretch as a measure of routing performance. The authors also thank the anonymous referees for their suggestions.

REFERENCES

ADAMIC, L., LUKOSE, R., PUNIYANI, A., AND HUBERMAN, B. 2001. Search in power-law networks. *Phys. Rev. E 64*, 4, 46135.

ARCAUTE, E., CHEN, N., KUMAR, R., LIBEN-NOWELL, D., MAHDIAN, M., NAZERZADEH, H., AND XU, Y. 2007. Deterministic decentralized search in random graphs. In *Proceedings of the 5th International Conference on Algorithms and Models for the Web-Graph*. 187–194.

BARRAT, A., BARTHELEMY, M., PASTOR-SATORRAS, R., AND VESPIGNANI, A. 2004. The architecture of complex weighted networks. *Proc. Nat. Acad. Sci. 101*, 11, 3747–3752.

BENDER, M. AND COLTON, M. 2000. The LCA problem revisited. In *Proceedings of the 4th Latin American Symposium on Theoretical Information (LATIN'00)*.

BRANDES, U. 2001. A faster algorithm for betweenness centrality. *J. Math. Sociol. 25*, 2, 163–177.

BRANDES, U. AND PICH, C. 2007. Centrality estimation in large networks. *Int. J. Bifurcation Chaos 17*, 7, 2303–2318.

CHOW, E. 2004. A graph search heuristic for shortest distance paths. Tech. rep., UCRL-JRNL-202894, Lawrence Livermore National Laboratory.

COHEN, E. AND ZWICK, U. 2001. All-pairs small-stretch paths. *J. Algor. 38*, 2, 335–353.

DABEK, F., COX, R., KAASHOEK, F., AND MORRIS, R. 2004. Vivaldi: A decentralized network coordinate system. In *Proceedings of the ACM SIGCOMM*. 15–26.

DIJKSTRA, E. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik 1*, 1, 269–271.

DOR, D., HALPERIN, S., AND ZWICK, U. 1996. All-pairs almost shortest paths. In *Proceedings of the Annual Symposium on Foundations of Computer Science 37*. 452–461.

ERDOS, P. AND RENYI, A. 1959. On random graphs. *Publ. Math. Debrecen 6*, 290.

FALOUTSOS, C., MCCURLEY, K., AND TOMKINS, A. 2004. Fast discovery of connection subgraphs. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 118–127.

FAST, A., FRIEDLAND, L., MAIER, M., TAYLOR, B., JENSEN, D., GOLDBERG, H., AND KOMOROSKE, J. 2007. Relational data preprocessing techniques for improved securities fraud detection. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 941–949.

FLAKE, G., TARJAN, R., AND TSIOUTSIOULIKLIS, K. 2004. Graph clustering and minimum cut trees. *Internet Math. 1*, 4, 385–408.

FORTUNATO, S. 2010. Community detection in graphs. *Physics Rep. 486*, 75–174.

FREEMAN, L. 1979. Centrality in social networks: Conceptual clarification. *Soc. Netw. 1*, 3, 215–239.

FRIEDLAND, L. AND JENSEN, D. 2007. Finding tribes: Identifying close-knit individuals from employment patterns. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 290–299.

GAVOILLE, C., PELEG, D., PERENNES, S., AND RAZ, R. 2001. Distance labeling in graphs. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*. 210–219.

GEISBERGER, R., SANDERS, P., AND SCHULTES, D. 2008. Better approximation of betweenness centrality. In *Proceedings of the 10th Workshop on Algorithm Engineering and Experiments*. SIAM.

GOLDBERG, A. V. AND HARRELSON, C. 2005. Computing the shortest path: A* search meets graph theory. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*. 156–165.

HADJICONSTANTINOU, E. AND CHRISTOFIDES, N. 1999. An efficient implementation of an algorithm for finding K shortest simple paths. *Netw. 34*, 2, 88–101.

HAREL, D. AND TARJAN, R. 1984. Fast algorithms for finding nearest common ancestors. *SIAM J. Comput. 13*, 2, 338–355.

KLEINBERG, J. 2000a. Navigation in a small world. *Nature 406*, 6798, 845.

KLEINBERG, J. 2000b. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*. 163–170.

KLEINBERG, J., SLIVKINS, A., AND WEXLER, T. 2009. Triangulation and embedding using small sets of beacons. *J. ACM 56*, 6, 32.

KOREN, Y., NORTH, S., AND VOLINSKY, C. 2007. Measuring and extracting proximity graphs in networks. *ACM Trans. Knowl. Discov. Data 1*, 3, 12.

KORF, R. 1985. Depth-first iterative-deepening: An optimal admissible tree search. *Artif. Intell. 27*, 1, 97–109.

KRIOUKOV, D. AND YANG, K. 2004. Compact routing on Internet-like graphs. In *Proceedings of 23rd IEEE Conference on Computer Communications*.

KUMAR, R., LIBEN-NOWELL, D., AND TOMKINS, A. 2006. Navigating low-dimensional and hierarchical population networks. In *Proceedings of the 14th Annual European Symposium on Algorithms*. 480–491.

KURUMIDA, Y., OGATA, T., ONO, H., SADAKANE, K., AND YAMASHITA, M. 2006. A generic search strategy for large-scale real-world networks. In *Proceedings of the 1st International Conference on Scalable Information Systems*.

LESKOVEC, J. 2008. Dynamics of large networks. Ph.D. thesis, Carnegie Mellon University.

LESKOVEC, J., KLEINBERG, J., AND FALOUTSOS, C. 2005. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 177–187.

LUA, E. K., GRIFFIN, T., PIAS, M., ZHENG, H., AND CROWCROFT, J. 2005. On the accuracy of embeddings for Internet coordinate systems. In *Proceedings of the 5th ACM IMC Internet Measurement Conference*. 125–138.

MAO, Y. AND SAUL, L. 2004. Modeling distances in large-scale networks by matrix factorization. In *Proceedings of the 4th ACM IMC Internet Measurement Conference*. 278–287.

NEWMAN, M. 2004a. Analysis of weighted networks. *Phys. Rev. E 70*, 5, 56131.

NEWMAN, M. 2004b. Detecting community structure in networks. *Europ. Phys. J. B-Condensed Matter 38*, 2, 321–330.

NG, T. AND ZHANG, H. 2002. Predicting Internet network distance with coordinates-based approaches. In *Proceedings of the 21st IEEE Conference on Computer Communications*.

PIAS, M., CROWCROFT, J., WILBUR, S., HARRIS, T., AND BHATTI, S. 2003. Lighthouses for scalable distributed location. In *Proceedings of the International Workshop on Peer-To-Peer Systems*.

POLLNER, P., PALLA, G., AND VICSEK, T. 2006. Preferential attachment of communities: The same principle, but a higher level. *Europhysics Lett. 73*, 3, 478–484.

POTAMIAS, M., BONCHI, F., CASTILLO, C., AND GIONIS, A. 2009. Fast shortest path distance estimation in large networks. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*.

PROVOST, F., JENSEN, D., AND OATES, T. 1999. Efficient progressive sampling. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

RATTIGAN, M., MAIER, M., AND JENSEN, D. 2006. Using structure indices for efficient approximation of network properties. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 357–366.

RATTIGAN, M., MAIER, M., AND JENSEN, D. 2007. Graph clustering with network structure indices. In *Proceedings of the 24th ICML International Conference on Machine Learning*. 783–790.

ROLI, A. 2005. On the impact of small-world on local search. In *Proceedings of the Advances in Artificial Intelligence (AI*IA'05)*. 13–24.

RUSSELL, S. AND NORVIG, P. 1995. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Inc. Upper Saddle River, NJ.

SHAVITT, Y. AND TANKEL, T. 2004. Big-bang simulation for embedding network distances in Euclidean space. *IEEE/ACM Trans. Netw. 12*, 6, 993–1006.

ŞIMŞEK, Ö. AND JENSEN, D. 2008. Navigating networks by using homophily and degree. In *Proceedings of the National Academy of Sciences 105*, 35, 12758.

SLIVKINS, A. 2007. Distance estimation and object location via rings of neighbors. *Distrib. Comput. 19*, 4, 313–333.

TANG, L. AND CROVELLA, M. 2003. Virtual landmarks for the Internet. In *Proceedings of the 3rd ACM IMC Internet Measurement Conference*. 143–152.

TANG, L. AND CROVELLA, M. 2004. Geometric exploration of the landmark selection problem. In *Proceedings of the 5th Passive and Active Measurement Workshop*.

TAURO, S., PALMER, C., SIGANOS, G., AND FALOUTSOS, M. 2001. A simple conceptual model for the Internet topology. In *Proceedings of the IEEE Global Telecommunications Conference 3*.

THORUP, M. AND ZWICK, U. 2005. Approximate distance oracles. *J. ACM 52*, 1, 1–24.

TRAVERS, J. AND MILGRAM, S. 1969. An experimental study of the small world problem. *Sociometry 32*, 4, 425–443.

WALSH, T. 1999. Search in a small world. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*. 1172–1177.

WASSERMAN, S. AND FAUST, K. 1994. *Social Network Analysis: Methods and Applications*. Cambridge University Press.

WATTS, D. AND STROGATZ, S. 1998. Collective dynamics of 'small-world' networks. *Nature 393*, 6684, 409–10.

WATTS, D., DODDS, P., AND NEWMAN, M. 2002. Identity and search in social networks. *Science 296*, 5571, 1302–1305.

WONG, B., SLIVKINS, A., AND SIRER, E. 2005. Meridian: A lightweight network location service without virtual coordinates. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. 85–96.

ZWICK, U. 2001. Exact and approximate distances in graphs—A survey. In *Proceedings of the 9th Annual European Symposium on Algorithms*. 33–48.

Received September 2009; revised February 2010, October 2010; accepted December 2010