A Survey on Proximity Measures for Social Networks

Sara Cohen[1], Benny Kimelfeld[2], and Georgia Koutrika[3]

[1] Dept. of Computer Science and Engineering
Hebrew University of Jerusalem
Jerusalem, Israel
sara@cs.huji.ac.il
[2] IBM Research–Almaden
San Jose, CA 95120, USA
kimelfeld@us.ibm.com
[3] HP Labs
Palo Alto, USA
koutrika@hp.com

Abstract. Measuring proximity in a social network is an important task, with many interesting applications, including person search and link prediction. Person search is the problem of finding, by means of keyword search, relevant people in a social network. In user-centric person search, the search query is issued by a person participating in the social network and the goal is to find people that are relevant not only to the keywords, but also to the searcher herself. Link prediction is the task of predicting new friendships (links) that are likely to be added to the network. Both of these tasks require the ability to measure proximity of nodes within a network, and are becoming increasingly important as social networks become more ubiquitous.

This chapter surveys recent work on scoring measures for determining proximity between nodes in a social network. We broadly identify various classes of measures and discuss prominent examples within each class. We also survey efficient implementations for computing or estimating the values of the proximity measures.

1 Introduction

Online social networks have grown in popularity at an extraordinary pace over the last few years. In fact, social networks, such as Facebook, MySpace and Twitter, have become so widespread that they currently boast hundreds of millions of users. The graph structure defined by a social network encodes interesting and useful information about the social relations between users. Leveraging this data to effectively answer different types of queries is an interesting and challenging problem.

Abstractly, a social network is simply a graph of people. Edges indicate that one person (node) likes/trusts/recommends another. This graph may be undirected (e.g., as in Facebook) or directed (e.g., as in Twitter). In some scenarios

S. Ceri and M. Brambilla (Eds.): Search Computing III, LNCS 7538, pp. 191–206, 2012.

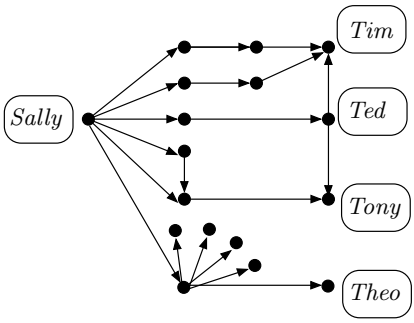

Fig. 1. Small fragment of a social network

edge weights are also desirable [44] to express the strength of a relationship, possibly by measuring the frequency of interactions between users. In addition, each node in a social network is typically associated with textual data, such as personal information, posts, etc.

The focus of this article is on surveying *proximity measures* for social networks. Intuitively, a proximity measure is a method of quantifying the degree of closeness between two given nodes s and t. Measuring proximity is an important aspect of several social network problems, including the following:

- *Person search* is the problem of finding, by means of keyword search, relevant people in a social network. Person search is an important type of query over a social network, as it is an aid in finding people of interest. In *user-centric person search*, the search query is issued by a person s participating in the social network, and the goal is to find people that possess two qualities: relevancy to the query, and relevancy to s herself. Proximity is highly important in ranking results of a user-centric person search, as nodes in close proximity to s are people (transitively) trusted by s. Hence, s can be less wary of entering into a real-life interaction (social or otherwise) with these people.
- *Link prediction* is the problem of predicting, using the graph structure of the network, the new relationships (i.e., edges) that are likely to be added in the near future. The ability to predict new edges for the network is useful for friend recommendation. It has been observed in the past that new social relations are likely to involve people who are already close one to another in a social network (e.g., triadic closure [37]). Thus, measuring proximity is an important aspect of solving the link prediction problem.

Now, consider the problem of measuring proximity in the social network of Figure 1. In this example, nodes have a name for easy reference. Suppose now that we would like to determine the proximity of Sally to each of the nodes Tim, Ted, Tony and Theo. This may occur as Sally poses a query for which these nodes are relevant (user-centric person search), or perhaps, we would like to

determine which of these nodes is most likely to enter a relationship with Sally (link prediction).

Devising a general proximity measure in a large network is a difficult task, since many different properties of the network should be taken into consideration. For example, a shorter path from s to t should increase the proximity of t to s. According to this measure, Ted is closer to Sally than Tim (as the distance from Sally to Ted is 2, while the distance from Sally to Tim is only 3). The existence of multiple paths from s to t should again increase the proximity of t. According to this measure, Tony would be closer to Sally than Ted. Additional graph features might also affect proximity, such as the disjointedness of paths from s to t and the out-degrees of nodes on the paths from s to t.

Trying to combine such features raises interesting questions about how the four nodes should be ranked. *Should Tim be ranked above or below Tony?* Tim has more disjoint paths from Sally, but the paths are longer. *Should Theo be ranked above or below Ted?* The paths from Sally to Ted and to Theo are of the same length, but the path from Sally to Theo is more "diluted" by nodes with high out-degrees than the path from Sally to Ted.

This article surveys recent work on proximity measures for social networks. As there has been considerable work introducing a variety of proximity measures, we do not profess to exhaustively cover all measures that have been proposed in the past. Instead, we broadly identify various classes of measures (by their underlying intuition and motivation). In Section 2, for each class, we discuss prominent examples of proximity measures. Next, Section 3 considers the problem of implementing these measures over huge social networks, and surveys recent work on efficiently computing or estimating proximity measures.

We note that this chapter differs greatly from previous works, such as that of Liben-Nowel and Kleinberg [32] that focus on comparing the effectiveness of various proximity measures for link prediction. First, our focus is on general proximity measures (some of which may be useful for search but not for link prediction). Second, we do not present an empirical comparison of the measures, as clearly different measures may be useful for different proximity-related tasks. Finally, we extensively survey implementations of the various proximity measures, which is pivotal for a real-world system.

2 Ranking Functions

A social network is a directed graph $G(V, E, txt)$, where V is a set of nodes, called people, $E \subseteq V \times V$ is a set of edges and for all $v \in V$, $txt(v)$ associates v with textual content, such as personal information, posts and so forth. Unless otherwise stated, social networks are directed, and thus, allow for asymmetric social relations. An edge (u, v) indicates that u views v in a positive light, that is, u likes/trusts/recommends v. A small fragment of a social network appears in Figure 1. Note that most textual content has been omitted in this figure for simplicity in presentation.

In this article, we will consider functions $\text{prox}(s, t, G)$ that measure the proximity of t to s in graph G. In these ranking functions, greater values are preferable,

that is, $\text{prox}(s, t, G) > \text{prox}(s, t', G)$ implies that t is, in some sense, closer to s than t' in G. Such a function can be used in user-centric person search, in order to rank query results, and in link prediction, in order to suggest likely new links.

In some contexts, the quality of a proximity ranking function can be measured. For example, datasets for measuring proximity with respect to link prediction are relatively easy to obtain (by recrawling a social network after some time has elapsed). However, there are no known benchmarks to test the quality of a proximity ranking function in the context of user-centric search. One might suppose that a proximity function proven to be superior for link prediction will also be superior for user-centric search. However, link prediction and user-centric search are quite different tasks. Conceptually, there is no reason to believe that a desired search result would also be a desired neighbor; indeed, an algorithm for link prediction may return none of the nodes matching a given set of keywords, as no corresponding link is predicted, and then provide no ranking at all for candidate answers. Moreover, while link prediction and benchmarks thereof typically capture relationships that already exist in real life, search is often about discovery beyond present knowledge.

Due to the lack of test data, and to the fact that different measures may be useful in different contexts, in this article we make no pretense of comparing proximity measures and pointing out which one(s) are of highest effectiveness. Instead we take a classification-based approach. Specifically, our discussion in this section is based on classifying proximity measures by their underlying intuition and motivation. Thus, we discuss measuring proximity based on shortest paths (Section 2.1), node neighborhoods (Section 2.2), random walks (Section 2.3), network flow (Section 2.4) and network sampling (Section 2.5). Finally, we discuss a property-based approach to classifying proximity measures (Section 2.6).

2.1 Shortest Paths

Certainly the simplest notion of measuring proximity is to use the length of the *shortest path* from s to t for this purpose. In order to have a greater proximity value for closer nodes, we can define this function, called $rdist(s, t, G)$, simply as the reciprocal of the distance from s to t in G, that is,

$$rdist(s, t, G) \stackrel{\text{def}}{=} (dist(s, t, G))^{-1} ,$$

where $dist(s, t, G)$ is the length of the shortest path from s to t. (An alternative is to use $-dist(s, t, G)$, instead of the reciprocal.)

The function $rdist(s, t, G)$ has the distinct advantage of being quite simple. Hence, efficiently computing this value over a dynamically changing and enormous social network is more feasible than for many of the other functions considered later. Computing $rdist(s, t, G)$ is discussed further in Section 3. On the other hand, $rdist(s, t, G)$ is oblivious to the structure of the social network, beyond edges on the shortest path. Hence, $rdist(s, t, G)$ is not very discriminating. As an example, Ted, Tony and Theo all have the same distance from Sally in Figure 1.

The *Katz measure* [29] captures the idea that the more paths that there are between s and t, and the shorter these paths are, the closer t is to s. Formally,

$$katz(s,t,G) \stackrel{\text{def}}{=} \sum_{l=1}^{\infty} \beta^l |\text{paths}^l_{s,t,G}|\,, \tag{1}$$

where $\text{paths}^l_{s,t,G}$ is the set of all length-l paths from s to t in G and the constant β dampens by length to count short paths more heavily.

Since *katz* takes all paths from s to t into consideration, it is much more discriminating that $rdist(s,t,G)$. In Figure 1, Ted and Theo have the same proximity to Sally, according to *katz* while Tony has higher proximity (due to the extra path of length 3 from Sally to Tony).

2.2 Node Neighborhoods

Several proximity measures are based on the neighborhoods of nodes s and t. We use $\Gamma(x)$ to denote the set of (undirected) neighbors of x. Preferential attachment [34] is the simplest neighborhood-based measure, and defines the similarity of s and t simply by $|\Gamma(s)| \times |\Gamma(t)|$. Thus, with respect to preferential attachment, a node s will always be most similar to nodes t with high degree. Preferential attachment has been successfully used to model the growth of networks and has been shown to be useful for link prediction in citation networks [6].

By considering not only the sizes of the neighborhoods of s and t, but also their intersection, we can further analyze the similarity of s and t. Intuitively, if s and t have many common neighbors, then they are likely to be closely related. Several neighborhood-based proximity measures were surveyed by Liben-Nowell and Kleinberg [32], including: the number of common neighbors, $|\Gamma(s) \cap \Gamma(t)|$, Jaccard coefficient of neighbors

$$\frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$$

and an adaptation of the Adamic/Adar measure [1]

$$\sum_{v \in \Gamma(s) \cap \Gamma(t)} \frac{1}{\log(|\Gamma(v)|)}\,.$$

Note that the last measure uses the log function to weight rarer features more heavily. A variation of the Adamic/Adar measure, that reduces the punishment on large-degree common neighbors, by defining similarity as

$$\sum_{v \in \Gamma(s) \cap \Gamma(t)} \frac{1}{|\Gamma(v)|}\,,$$

has been introduced [50] and has been shown to outperform Adamic/Adar on some networks.

The above proximity measures have proven useful for link prediction [32], especially since links tend to be added between close nodes. However these measures are inappropriate for person search. In particular all neighborhood-based measures (other than preferential attachment), give a score of 0 if s and t have no common neighbors. For person search, relevant answers may be much farther from s. Thus, it is even possible that no node within distance two from s even satisfies the textual part of the person query, and hence all nodes relevant to the query might receive a rank of zero using neighborhood-based proximity measures.

2.3 Random Walks

Ranking measures based on random walks, such as PageRank, have proven extraordinarily successful for search on the Web. Intuitively, the PageRank of a Web page t is a measure of the likelihood that a Web surfer, starting at a random page, randomly choosing outgoing links to click on (and occasionally teleporting to a random Web page), will reach t. More precisely, the PageRank of t is the stationary probability of t in a random walk that jumps to a random node in the graph with probability α at each step, and moves to a random neighbor with probability $1 - \alpha$.

PageRank is not immediately applicable to ranking proximity, as it measures the importance of a page t in the entire graph, and not with respect to another node s. A relevant adaptation of PageRank is that of *personalized PageRank* [26]. In this ranking function, there is a probability distribution Ω over the nodes of the graph. The personalized PageRank of t is the stationary probability of t in a random walk that jumps to a random node v in the graph with probability $\alpha \times \Omega(v)$ at each step, and moves to a random neighbor with probability $1 - \alpha$. Thus, personalized PageRank favors nodes with high probability according to Ω (as well as their neighbors, and neighbors' neighbors, etc.).

Rooted PageRank [31, 32], a special case of personalized PageRank, was considered for measuring proximity in a social network (in the context of link prediction). In rooted PageRank, there is a designated node s, called the root, for which $\Omega(s) = 1$, while $\Omega(v) = 0$, for all other nodes $v \neq s$. Thus, rooted PageRank, denoted $rPR(s, t, G)$, is the stationary probability of t in a random walk that *returns to* s with probability α at each step, moving to a random neighbor with probability $1 - \alpha$. Intuitively, this function gives a higher ranking to nodes t that are more easily reached from s when traversing G.

2.4 Flow in Networks

Koren et al. [30] developed sophisticated proximity measures for an *undirected and weighted* graph. They propose that proximity should be more sensitive to edges between low-degree nodes that show meaningful relationships and should take into account multiple paths between s and t. To define such a measure, they first consider *network flow* as a ranking function. The network flow from s to t grows as the number of paths from s to t increases, as intuitively required from a

proximity measure. However, network flow is not sensitive to path lengths, and is bounded by the s-t-cut[1] capacity, both of which are undesirable.

To overcome the problems associated with network flow, Koren et al. [30] model the network as an electrical circuit. Intuitively, edges can be seen as resistors, with s having a voltage of 1, and t having a voltage of 0. Then, a series of linear equations can be used to estimate the currents of the network, and in particular, the current delivered from s to t, called the *effective conductance* [13]. They proposed *cycle-free effective conductance* as an improvement over effective conductance for measuring network proximity. This measure is quite intricate, and we present the main details here. (See [30] for full details.)

In a random walk, the probability to follow an edge from node u to node v is $w_{u,v}/\deg_u$ where $w_{u,v}$ is the weight of the edge from u to v and $\deg_u$ is the degree of u. Thus, the probability of a random walk following the path $\bar{v} = v_1, \ldots, v_n$ is simply the product of the probabilities of each transition in the path. The *cycle-free escape probability* from s to t is the probability that a random walk beginning at s will reach t without visiting any node more than once. Finally, the proposed measure of proximity, called *cycle-free effective conductance* is the product of the degree of s and the cycle-free escape probability from s to t.

2.5 Random Sampling

A rather different approach to measuring proximity comes from the field of communication networks, where the notion of *network reliability* was considered [10, 20]. In this measure, there is a fixed probability $p \in (0, 1)$. We denote by G^{r} a random subgraph of G that is obtained by removing each edge of G, independently, with probability $1-p$. The reliability of G is the probability that G^{r} is connected. The greater the reliability of a network, the more likely it is that communication will be possible with all nodes, even in the presence of network failures.

Reliability is a global function of a graph. However, *two-terminal network reliability* [41] is the natural counterpart of this function for a given pair of nodes s and t. Thus, two-terminal network reliability measures the likelihood that there will be a path from s to t in a random subgraph, formally defined as

$$\mathit{reliability}_2(s, t, G) \stackrel{\text{def}}{=} \Pr\left[G^{\mathrm{r}} \text{ has a path from } s \text{ to } t\right] .$$

Obviously, the closer t is to s, and the more (independent) paths there are from s to t, the higher the two-terminal network reliability will be. This function has been considered for proximity ranking in user-centric person search [9].

Another sampling based function considered for person search [9] is that of *expected distance*. Intuitively, this function measures the expected distance from s to t, when each edge is removed with probability $1-p$. Note that for this value

[1] Recall that an s-t-cut is a partition of the graph into two disjoint sets of nodes, one of which contains s and the other of which contains t. The capacity of the cut is the sum of weights of edges "split" by the cut.

to be well defined, there must be a number m, that is returned by the function, when no path from s to t exists.

Once again, we assume that there is a fixed probability $p \in (0, 1)$. In addition, we fix a parameter $m \in \mathbb{R}$. We will implicitly assume that m is larger than the number of nodes in the graph G. The *m-bounded distance* from s to t, denoted $\hat{\delta}_G(s, t)$, is defined by

$$\hat{\delta}_G(s, t) \stackrel{\text{def}}{=} \min\{dist(s, t, G), m\} .$$

Thus, if G has no path from s to t, then $\hat{\delta}_G(s, t) = m$. Note that if $s \neq t$, then $\hat{\delta}_G(s, t)$ is always in the interval $[1, m]$.

We denote by G^{r} a random subgraph of G that is obtained by removing each edge of G, independently, with probability $1 - p$. The *expected* m-bounded distance, denoted by $\overline{\delta_G}(s, t)$, is defined as follows.

$$\overline{\delta_G}(s, t) \stackrel{\text{def}}{=} \mathbb{E}\left[\hat{\delta}_{G^{\mathrm{r}}}(s, t)\right] .$$

That is, $\overline{\delta_G}(s, t)$ is the expected m-bounded distance from s to t in a random subgraph G^{r} of G. Finally, the proposed proximity ranking function is the reciprocal of $\overline{\delta_G}(s, t)$, namely

$$expd(s, t, G) \stackrel{\text{def}}{=} (\overline{\delta_G}(s, t))^{-1} .$$

2.6 Properties for Proximity Ranking Functions

In lieu of empirical comparison of proximity functions (which is not always currently possible due to lack of benchmarks), Cohen et al. [9] propose three simple properties that proximity functions $\mathrm{prox}(s, t, G)$ should satisfy. They analyze a variety of functions with respect to these properties. Intuitively, the properties are based on the observation that certain graph transformations should only increase $\mathrm{prox}(s, t, G)$, as these transformations, in a sense, make the relationship between s and t more significant. In a way, this is similar in spirit to the underlying premise of the family of TF-IDF ranking functions [33] for textual search and ranking. This premise requires that increasing term occurrences within a document, or decreasing term frequency within a corpus, should only increase the ranking of a document with respect to the given term.

One property of Cohen et al. [9] requires the following. Suppose that a node v lies on a simple path from s to t in G. Moreover, suppose that v has a single incoming and outgoing edge. Then, removing v from G (and directly connecting its incoming and outgoing neighbors) can only shorten paths from s to t without having additional effect on the graph. Thus, such a transformation should only cause $\mathrm{prox}(s, t, G)$ to grow. Other properties of Cohen et al. [9] consider the effect expected when paths from s to t become more disjoint. These properties state that (under certain conditions) splitting a node into several nodes (while preserving existing paths) should again only increase $\mathrm{prox}(s, t, G)$. We demonstrate these ideas in the following example.

Example 1. Let G be the graph of Figure 1, and let s be the node Sally. Removing the node on the path from Sally to Ted should raise Ted's score. Splitting the node with two incoming edges on the paths from Sally to Tony would result in the graph structure containing two disjoint paths from Sally to Tony (with lengths of two and three) and hence, should raise Tony's score.

Cohen et al. [9] analyzed the satisfaction of the given properties by different proximity measures, namely, shortest path, the Katz measure, rooted PageRank, reliability and expected distance. They showed that, of the measures considered, only those based on random sampling (reliability and expected distance) satisfied all properties *in the strong sense* (i.e., the graph transformation guaranteed the ranking the increase). As one example where other measures failed to strongly satisfy all properties, note that the Katz measure is oblivious to disjointness of paths. Hence, this measure does not increase given graph transformations making paths more disjoint. (For full details see [9].)

3 Efficiently Computing or Estimating Ranking Functions

Real-life social networks are often huge, easily containing hundreds of millions of members. Storing pairwise ranking values (for any chosen ranking function) is infeasible, due to its huge memory requirements. In addition, the dynamic nature of social networks, which are constantly changing and evolving, would seem to quickly make pre-computed ranking values obsolete. Therefore, answering person search queries or link prediction requires online computation (or estimation) of ranking functions. In this section, we reconsider the ranking functions introduced in Section 2, and survey recent algorithms for their efficient computation, or estimation.[2]

3.1 Shortest Paths

The first ranking function considered, $rdist(s, t, G)$, simply computes the reciprocal of the shortest distance from s to t. Thus, to compute this function, an algorithm for finding shortest paths in needed. Due to the simplicity of this function, and to the many applications using shortest paths, it is not surprising that this is the most well-studied ranking function.

Computing shortest paths is a well-studied problem, with many well-known solutions. For example, the single-source shortest path problem can be solved using breadth-first search in $O(|V| + |E|)$ for unweighted graphs, and can be solved using Dijkstra's algorithm [11] in time $O(|V|^2)$ (or $O(|V|log|V| + |E|)$ for sparse graphs) for weighted graphs. The all-pairs shortest paths problem can be solved using the Floyd-Warshall algorithm in time $O(|V|^3)$ [17]. It would therefore seem that this ranking function needs no further treatment.

[2] We do not discuss implementing neighborhood based proximity measures as these are typically straightforward.

The response time for a search is typically expected to be within a few milliseconds. However, in practice, the online computation of the above algorithms in huge social networks is simply too slow. For example, Potamias et al. [36] experimentally show that a standard PC requires a minute to compute a full breadth-first search (BFS) traversal of a network containing only four million nodes and 50 million edges. Hence, recent work has focused on significantly speeding up processing time by approximating the shortest path length, instead of its precise computation.

Several different methods have been introduced to estimate shortest path distances. One method is to choose a subset of the nodes, called *landmarks* [36] or *seeds* [42]. Instead of computing all-pairs shortest paths, the shortest path from each landmark to every other node in the graph is pre-computed and stored. When the distance from s to t is desired, this value is estimated using the distance of s and t to each of the landmarks. (In particular, the distance of s to t is at most the minimal sum of the distance of s to any landmark u, and the distance of u to t).

An interesting question is how to best choose the landmarks, so as to derive a good estimation of distances. To be precise, a set of landmarks *covers* a pair of vertices s, t if there exists at least one landmark in the set which lies on a shortest path from s to t. (Using such a landmark will yield the precise distance from s to t.) Potamias et al. [36] showed that selecting k landmarks so as to maximize the number of covered pairs is an NP-complete problem. However, they have presented and experimentally studied various strategies for landmark selection, and have shown their strategies to be quite effective and efficient in practice. Adding small path sketches to the information stored at the landmarks has been considered to allow shortest paths (and not just their lengths) to be computed and retrieved [23].

Efficiently finding the precise distance between two nodes (called point-to-point shortest paths) was studied by Goldberg and Harrelsons [22]. There, precomputed landmarks are used as an upped bound for the actual distance. These bounds are leveraged to compute the exact distance, based on A^* search and using the triangle inequality, thereby defining a new class of algorithms, called *ALT algorithms* (for A^*, landmarks and triangle inequality). They show significant improvement on the number of neighbors traversed during computation, with respect to the state of the art.

Landmarks have also been used as a bootstrapping stage for *graph coordinate systems* [48, 49]. Intuitively, a graph coordinate system maps the nodes of a graph to coordinates in a Euclidean or hyperbolic space. Once such a mapping is available, shortest path estimation is easily achieved, by simply computing the distance between the coordinates corresponding to the nodes of interest. However, finding an effective mapping is a difficult problem.

The system architecture can have a significant impact on the speed of computing shortest paths. Katz and Kider [28] investigated the problem of computing all-pairs shortest paths on a Graphics Processing Unit (GPU). They present a highly parallel and scalable formulation of a transitive closure, and then use this

to run the Floyd-Warshall algorithm on a GPU. A significant speedup is shown in comparison to runtime on a CPU.

Recently, Gao et al. [21] considered leveraging a relational database to compute shortest paths. For that purpose, they introduce the *FEM framework* with new operators that are suitable for the task at hand. Optimizations (such as use of new SQL features, and bi-directional set Dijkstra) are presented to further speed-up shortest path calculation. Finally, Xiao et al. [45] consider the problem of pre-indexing all shortest path distances. As storing all distances is not feasible, due to the amount of memory required, they focus on reducing the required memory size. To achieve this, they exploit graph symmetry (using graph automorphisms), and enable indexing at the *orbit level* instead of at the *node level*.

3.2 Katz Measure

While shortest path considers a single path from source to target, the Katz measure is a function of *all* source-to-target paths. Obviously, this makes computing, or estimating, $katz(s, t, G)$ a much more difficult task. There are very few works attempting to solve this problem.

One method of speeding up the computation of the Katz measure is to truncate the computation at paths of a specific, predetermined, length. Thus, the sum of Equation (1) will be computed only up to this predefined length (and not until infinity) [19, 43]. The wealth of work on enumerating shortest paths (e.g., [7,14,27,46,47]) can be leveraged to compute the truncated Katz measure.

Two new techniques, called *proximity sketches* and *proximity embeddings* were introduced by Song et al. [39] to efficiently estimate a family of proximity measures. Interestingly, they show that the Katz measures, rooted PageRank and escape probability can all be estimated efficiently, if the *proximity inversion problem* can be efficiently solved. The basic idea is that all three measures can be defined as functions of the adjacency matrix of the network, and thus, can be computed by matrix inversion. Proximity sketches and embeddings are introduced as dimension reduction techniques, so as to make sparser the matrix that must be inverted, and hence, allow its inversion to be efficiently achieved.

While Song et al. [39] compute the Katz measure for all pairs, Esfandiar et al. [15] focus on computing pairwise $katz(s, t, G)$, in order to reduce computation time. They introduce a technique that combines Lanczos iteration and a quadrature rule to compute the values of interest. (Their work can also be used to compute another proximity function, called *commute time*.)

3.3 Rooted PageRank

As discussed earlier, rooted PageRank is a special type of personalized PageRank ranking function, which, in turn, adapts the classical PageRank by changing its teleporting mechanism. In the previous section we discussed one method to efficiently estimate PageRank [39]. In this section, we consider other methods that have been developed for estimating personalized PageRank. We note that

work on computing PageRank generally takes one of two approaches: using linear algebraic techniques or Monte Carlo methods. Unsurprisingly, this is also the case with personalized PageRank.

We start by discussing work that use *linear algebraic techniques* (as does the simple power iteration method for computing PageRank [35]). Jeh and Widom [26] explored the computation of personalized PageRank in a scalable manner. One of the crucial aspects of their algorithm is the assumption that the *preference set* (i.e., nodes to which teleporting is allowed) is always a subset of a given set of nodes H (called hubs). Unfortunately, for rooted PageRank, the set H is precisely all nodes in the network. Hence, the method presented of Jeh and Widom [26] reduces to a simple dynamic programming algorithm that provides no performance improvement over the standard power iteration method [35]. Sarlós et. al [38] improve upon that method by using *deterministic rounding* and *randomized sketching* techniques. Thus, their approach is for unrestricted on-line personalized PageRank.

In *Monte Carlo methods*, the basic idea is to approximate PageRank by directly simulating the corresponding random walks and then estimating the stationary distributions with the empirical distributions of the performed walks. Based on this idea, the following method for approximating personalized PageRank has been proposed [2, 18]. Starting at each node u, perform a number R of random walks, called *fingerprints*, each having a length geometrically distributed. They have shown that the frequencies of visits to different nodes in these fingerprints will approximate personalized PageRanks. Monte Carlo algorithms to compute personalized PageRank have also been studied [3, 4].

3.4 Effective Conductance

After introducing cycle-free effective conductance (CFEC), Koren et al. [30] provide an efficient method for approximating this value. In their approximation, they use only the most probable paths between a source node s and a target node t. The authors have experimentally found that path probability falls off exponentially, hence the low probability paths cannot sum to any significant value. Therefore, the k most probable simple paths are determined by some threshold. This is similar, in a sense, to the notion of truncating the Katz measure, discussed earlier.

Based on the above idea, the problem of cycle-free effective conductance estimation is mapped to the k shortest-simple-paths problem, a natural generalization of the shortest path problem, in which not one but several paths in order of increasing length are sought. They employ an algorithm for the computation of the k most probable paths [25] for the CFEC estimation. Typically, the algorithm stops when the probability of the unscanned paths drops significantly below that of the most probable path (e.g., below a factor of 10^{-6}).

3.5 Reliability and Expected Distance

The problem of computing expected distance seems to have not been studied in the past. Hence, its complexity, as well as methods for estimating this value, are currently unknown. Therefore, in this section we focus only on computing and estimating two-terminal network reliability.

It was shown [41] that exact calculation of two-terminal reliability for general networks is #P-complete. (Recall that #P is the class of the counting problems associated with the decision problems in NP.) Hence, precisely computing two-terminal reliability is likely to be highly intractable. Instead, there has been work focusing on computing upper and lower bounds to this measure, while avoiding the exponential computation likely to required by exact algorithms.

The difficulty of two-terminal reliability and its many interesting applications in networks have stimulated many different approaches to estimating two-terminal reliability. These include partitioning techniques [12], techniques based on the sum of disjoint products [5], and Monte-Carlo simulations [16]. Terruggia [40] surveys these approaches.

4 Conclusion

In this article we surveyed a variety of proximity measures for social networks. We also discussed the underlying principles guiding the development of these proximity measures (e.g., based on shortest paths, sampling, flow, random walks). Efficient algorithms for computing or estimating proximity measures were also surveyed. The tradeoffs of simplicity versus efficient computability are clearly apparent.

Only graph-based similarity measures were considered in this article. However, there additional types of information can be useful for determining similarity of nodes, when available. For example, [8] leverages tags on nodes for people search, and [24] takes a crowd-sourcing approach to determining node similarity.

There are many related problems that are still open. Development of benchmark data for user-centric person search is an important problem. Only such a benchmark can guide the development of proximity measures for user-centric person search. We also observe that many of the algorithms for computing (or estimating) proximity measures do not adapt well to changes in the social network. Since social networks are constantly changing and evolving, this is a critical issue. Finally, it would be interesting to adapt the proximity measures we discussed to weighted graphs, and to graphs with edge labels that represent different kinds of relationships such as “follower,” “friend of” and “spouse of,” and even ones carrying a negative sentiment like “warns about” and “denounces.”

Acknowledgements. The research of Sara Cohen was partially supported by the ISF (Grant 143/09) and the Israeli Ministry of Science and Technology (Grant 3-6472).

References

1. Adamic, L., Adar, E.: How to search a social network. In: VLDB, pp. 217–225 (1987)
2. Avrachenkov, K., Litvak, N., Nemirovsky, D., Osipova, N.: Monte carlo methods in pagerank computation: When one iteration is sufficient. SIAM J. Numer. Anal. 45(2), 890–904 (2007)
3. Bahmani, B., Chakrabarti, K., Xin, D.: Fast personalized pagerank on mapreduce. In: SIGMOD Conference, pp. 973–984 (2011)
4. Bahmani, B., Chowdhury, A., Goel, A.: Fast incremental and personalized pagerank. PVLDB 4(3), 173–184 (2010)
5. Balan, A.O., Traldi, L.: Preprocessing minpaths for sum of disjoint products. IEEE Trans. Reliability 52(3), 289–295 (2003)
6. Barabasi, A.L., Jeong, H., Neda, Z., Ravasz, E., Schubert, A., Vicsek, T.: Evolution of the social network of scientific collaborations. Physica A 311(3-4), 590–614 (2002)
7. Brander, A., Sinclair, M.: A comparative study of k-shortest path algorithms. In: Proc. 11th UK Performance Engineering Workshop for Computer and Telecommunications Systems (1995)
8. Carmel, D., Zwerdling, N., Guy, I., Ofek-Koifman, S., Har'el, N., Ronen, I., Uziel, E., Yogev, S., Chernov, S.: Personalized social search based on the user's social network. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, pp. 1227–1236. ACM, New York (2009), http://doi.acm.org/10.1145/1645953.1646109
9. Cohen, S., Kimelfeld, B., Koutrika, G., Vondrák, J.: On principles of egocentric person search in social networks. In: First International Workshop on Searching and Integrating New Web Data Sources, Seattle, Washington (2011)
10. Davies, D., Barber, D.: Communication Networks for Computers. John Wiley, London (1973)
11. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271 (1959)
12. Dotson, W.P., Gobien, J.O.: A new analysis technique for probabilistic graphs. IEEE Trans. Circuits and Systems 26(10), 855–865 (1979)
13. Doyle, P., Snell, J.: Random walks and electical networks. The Mathematical Association of America (1984)
14. Eppstein, D.: Finding the k shortest paths. SIAM J. Comput. 28(2), 652–673 (1998)
15. Esfandiar, P., Bonchi, F., Gleich, D.F., Greif, C., Lakshmanan, L.V.S., On, B.-W.: Fast Katz and Commuters: Efficient Estimation of Social Relatedness in Large Networks. In: Kumar, R., Sivakumar, D. (eds.) WAW 2010. LNCS, vol. 6516, pp. 132–145. Springer, Heidelberg (2010)
16. Fishman, G.S.: A comparison of four monte carlo methods for estimating the probability of s-t connectedness. IEEE Trans. Reliability 35(2), 145–155 (1986)
17. Floyd, R.W.: Algorithm 97: Shortest path. Communications of the ACM 5(6), 345 (1962)
18. Fogaras, D., Rácz, B.: Towards Scaling Fully Personalized PageRank. In: Leonardi, S. (ed.) WAW 2004. LNCS, vol. 3243, pp. 105–117. Springer, Heidelberg (2004)
19. Foster, K.C., Muth, S.Q., Potterat, J.J., Rothenberg, R.B.: A faster katz status score algorithm. Computational and Mathematical Organization Theory 7, 275–285 (2001)
20. Frank, H., Frisch, I.: Communication, Transmission and Transportation Networks. Addison Wesley, Reading (1971)

21. Gao, J., Jim, R., Zhou, J., Yu, J.X., Jiang, X., Wang, T.: Relational approach for shortest path discovery over large graphs. PVLDB 5(4), 358–369 (2012)
22. Goldberg, A., Harrelsons, C.: Computing the shortest path: A* search meets graph theory. In: SODA (2005)
23. Gubichev, A., Bedathur, S.J., Seufert, S., Weikum, G.: Fast and accurate estimation of shortest paths in large graphs. In: CIKM, pp. 499–508 (2010)
24. Guy, I., Perer, A., Daniel, T., Greenshpan, O., Turbahn, I.: Guess who?: enriching the social graph through a crowdsourcing game. In: Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems, CHI 2011, pp. 1373–1382. ACM, New York (2011), http://doi.acm.org/10.1145/1978942.1979145
25. Hadjiconstantinou, E., Christofides, N.: An efficient implementation of an algorithm for finding k shortest simple paths. Networks 34, 88–101 (1999)
26. Jeh, G., Widom, J.: Scaling personalized web search. In: Proceedings of the 12th International Conference on World Wide Web, WWW 2003, pp. 271–279. ACM, New York (2003), http://doi.acm.org/10.1145/775152.775191
27. Katoh, N., Ibaraki, T., Mine, H.: An efficient algorithm for k shortest simple paths. Networks 12 (1982)
28. Katz, G.J., Kider Jr., J.T.: All-pairs shortest-paths for large graphs on the gpu. In: Graphics Hardware, pp. 47–55 (2008)
29. Katz, L.: A new status index derived from sociometric analysis. Psychometrika 18(1), 39–43 (1953)
30. Koren, Y., North, S.C., Volinsky, C.: Measuring and extracting proximity graphs in networks. TKDD 1(3) (2007)
31. Liben-Nowell, D., Kleinberg, J.M.: The link-prediction problem for social networks. In: CIKM (2003)
32. Liben-Nowell, D., Kleinberg, J.M.: The link-prediction problem for social networks. JASIST 58(7), 1019–1031 (2007)
33. Manning, C.D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press, New York (2008)
34. Mitzenmacher, M.: A brief history of generative models for power law and lognormal distributions. Internet Mathematics 1(2), 226–251 (2004)
35. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Tech. rep., Stanford University (1998)
36. Potamias, M., Bonchi, F., Castillo, C., Gionis, A.: Fast shortest path distance estimation in large networks. In: CIKM (2009)
37. Rapoport, A.: Spread of information through a population with socio-structural bias i: Assumption of transitivity. Bulletin of Mathematical Biophysics 15(4), 523–533 (1953)
38. Sarlós, T., Benczúr, A.A., Csalogány, K., Fogaras, D., Ráz, B.: To randomize or not to randomize: space optimal summaries for hyperlink analysis. In: World Wide Web, pp. 297–306 (2006)
39. Song, H.H., Cho, T.W., Dave, V., Zhang, Y., Qiu, L.: Scalable proximity estimation and link prediction in online social networks. In: IMC (2009)
40. Terruggia, R.: A comparison of four monte carlo methods for estimating the probability of s-t connectedness. Thesis. Università degli Studi di Torino (2010)
41. Valiant, L.G.: The complexity of enumeration and reliability problems. SIAM J. Comput. 8(3), 410–421 (1979)
42. Vieira, M.V., Fonseca, B.M., Damazio, R., Golgher, P.B., de Castro Reis, D., Ribeiro-Neto, B.A.: Efficient search ranking in social networks. In: CIKM, pp. 563–572 (2007)
43. Wang, C., Satuluri, V., Parthasarathy, S.: Local probabilistic models for link prediction. In: ICDM, pp. 322–331 (2007)

44. Xiang, R., Neville, J., Rogati, M.: Modeling relationship strength in online social networks. In: Proceedings of the 19th International Conference on World Wide Web, WWW 2010, pp. 981–990. ACM, New York (2010), http://doi.acm.org/10.1145/1772690.1772790
45. Xiao, Y., Wu, W., Pei, J., Wang, W., He, Z.: Efficiently indexing shortest paths by exploiting symmetry in graphs. In: EDBT (2009)
46. Yen, J.Y.: Finding the k shortest loopless paths in a network. Management Science 17 (1971)
47. Yen, J.Y.: Another algorithm for finding the k shortest loopless network paths. In: Proc. of 41st Mtg. Operations Research Society of America 20 (1972)
48. Zhao, X., Salaa, A., Wilson, C., Zheng, H., Zhao, B.Y.: Orion: Shortest path estimation for large social graphs. In: Proceedings of the 3rd Workshop on Online Social Networks, WOSN (2010)
49. Zhao, X., Salaa, A., Zheng, H., Zhao, B.Y.: Efficient shortest paths on massive social graphs. In: Proceedings of 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom (2011)
50. Zhou, T., Lü, L., Zhang, Y.C.: Predicting missing links via local information. The European Physical Journal B—Condensed Matter and Complex Systems 71(4), 623–630 (2009)