Measuring Graph Proximity with Blink Model

Haifeng Qian, Hui Wan, Mark N. Wegman, Luis A. Lastras, Ruchir Puri
IBM T. J. Watson Center, Yorktown Heights, NY
qianhaifeng,hwan,wegman,lastrasl,ruchir@us.ibm.com

ABSTRACT

This paper proposes a new graph proximity measure. This measure is a derivative of network reliability. By analyzing its properties and comparing it against other proximity measures through graph examples, we demonstrate that it is more consistent with human intuition than competitors. A new deterministic algorithm is developed to approximate this measure with practical complexity. Empirical evaluation by two link prediction benchmarks, one in coauthorship networks and one in Wikipedia, shows promising results. For example, a single parameterization of the proposed measure achieves accuracies that are 14–35% above the best accuracy for each graph of all predictors reported in the 2007 Liben-Nowell and Kleinberg survey.

1. INTRODUCTION

Humans have intuitions for graph proximity. Given a graph, certain pairs of nodes are perceived to have stronger relation strength than others. We know that a larger number of shorter paths indicate greater proximity, yet a precise mathematical formulation of such perception is elusive. Many measures have been defined in the literature that can be viewed as quantitative proxies of graph proximity: shortest path, Jaccard's coefficient [13], Katz [17], personalized PageRank [24], SimRank [14], Adamic/Adar [1] and others [4, 5, 8, 11, 19, 20, 25, 29, 31]. Although they each have characteristics that suit specific applications, they generally have varying degrees of agreement with human intuition.

This manuscript adds one more entry to the list. This graph proximity measure is called the Blink Model and is a derivative of network reliability. By studying its properties and a series of graph examples, we argue that it matches human intuition better than many existing measures. We develop a practical algorithm to approximately compute this measure, and demonstrate its predicting power through empirical validations. Some of the contents appeared in [27].

Relational data, or graph-structured data, are ubiquitous. Graph proximity measures, i.e., the ability to quantify relation strength, are fundamental building blocks in many applications. They can be used to recommend new contacts in social networks [21], to make product recommendations based on a graph model of products and users [2], to rank web search results or documents in general [24], or to predict new facts in knowledge graphs [23]. They can also be used to single out anomalies by identifying implausible links. The list of applications goes on and on.

The proposed Blink Model measure is a derivative of terminal network reliability [3] and is closely related to random graphs [12], including percolation theory [6] and uncertain graphs [15, 22, 25], and the independent cascade model [18] in network influence works. Network reliability has largely been ignored as a candidate measure in the aforementioned applications. For example, [25] concluded that network reliability was one of the least predictive measures. We prove the opposite conclusion with our Blink Model measure by including the winning measure from [25] in both theoretical studies and empirical validations. We attribute the discrepancy to the use of only 50 Monte Carlo samples in [25].

Exact evaluation of the Blink Model measure has the same complexity as terminal network reliability, which is known to be #P-complete [30]. Most methods from the field of network reliability [3, 7, 28] have poor scalability, and Monte Carlo has been considered the choice for larger and general graphs [10, 16]. We will present a new deterministic algorithm that approximates the proposed measure directly with practical complexity and thereby enables the proposed measure in applications.

To quantify the benefit of being consistent with human intuition, we use two link prediction tasks to compare our measure against other topological proximity measures. The first is a replication of [21]. A remarkable conclusion of [21] was that the best proximity measure is case dependent. A specific parameterization of a specific measure may perform well on one graph yet underperform significantly on another, and there does not exist a consistent winner. We compare against the oracle, i.e. the highest accuracy for each graph of all predictors in [21], and demonstrate that a single parameterization of our measure outperforms the oracle by 14–35% on each graph. The second task is predicting additions of inter-wikipage citations in Wikipedia from April 2014 to March 2015, and again substantial accuracy advantage is shown. We also demonstrate a simple yet practical and automatic method of training graph weighting parameters.

1.1 Problem statement

The problem statement for a graph proximity measure is the following. The input is a graph $G = \langle V, E\rangle$, its node weights $w_V : V \to (0, 1]$, and its edge weights $w_E : E \to (0, 1]$. The output is, for any pair of nodes A and B in V, a value score(A,B).

Note that not all proximity measures consider w_V and w_E. Some use w_E and ignore w_V, while some consider only topology G. Although w_V and w_E can be from any source, we present a simple yet practical method in Section 2.4 to

MLG '16 San Francisco, CA, USA
© 2016 ACM.

train a few parameters and thereby set w_V and w_E. It's applicable to all proximity measures and is used in Section 4.

For clarity, we will focus on directed simple edges. Undirected edges can be represented by two directed edges, and most discussions are extendable to hyperedges.

1.2 Definitions

Let us first define the proposed graph proximity measure. Consider the input G as a graph that blinks: an edge exists with a probability equal to its weight; a node exists with a probability equal to its weight. Edges and nodes each blink independently. A path is considered existent if and only if all edges and all intermediate nodes on it exist; note that we do not require the two end nodes to exist. The proposed proximity measure is

$$\begin{aligned} s(\mathrm{A},\mathrm{B}) &= -\log\left(1-b(\mathrm{A},\mathrm{B})\right) && (1)\\ \text{where}\quad b(\mathrm{A},\mathrm{B}) &= \mathrm{P}\left[\text{ a path exists from A to B }\right] && (2)\end{aligned}$$

We will refer to (1) as the Blink Model measure, its properties and generalizations to be presented in Section 2. It is straightforward to see that s and b are monotonic functions of each other and hence order equivalent, and the reason to choose s over b will be evident in Section 2.1.

Next we define several competing measures. For brevity, SimRank [14] and commute-time [11] are omitted and they are compared in Section 2.3.

Personalized PageRank (PPR) [24] with weights considers a Markov chain that has the topology of G plus a set of edges from each node to node A. These additional edges all have transition probability of $\alpha \in (0,1)$. For each original edge $e \in E$, let X be its source node, let $w_{\mathrm{sum,X}}$ be the sum of weights of X's out-going edges, and the transition probability of edge e is $(1-\alpha)\cdot w_E(e)/w_{\mathrm{sum,X}}$. The PPR measure $\mathrm{score}_{\mathrm{PPR}}$ (A,B) is defined as this Markov chain's stationary distribution on node B. PPR does not use node weights.

The original Katz measure [17] does not use edge or node weights, and we define a modified Katz measure which does:

$$\mathrm{score}_{\mathrm{Katz}}(\mathrm{A,B}) = \sum_{l=1}^{\infty}\left(\beta^l \cdot \sum_{\text{length-}l \text{ A-to-B path } i} p_{i,l}\right) \qquad (3)$$

where $\beta \in (0,1)$ is a parameter, and $p_{i,l}$ is the product of edge weights and intermediate node weights for the i^{th} path with length l. This measure is divergent if β is larger than the reciprocal of the spectral radius of the following matrix M: entry $M_{i,j}$ is the product of the i^{th} node weight and the sum of edge weights from the i^{th} node to the j^{th} node.

The effective conductance (EC) measure is defined as the effective conductance between two nodes by viewing edges as resistors. It can be generalized to be a directed measure, and notable variants include cycle-free effective conductance (CFEC) [19] and EC with universal sink [9, 29].

Expected Reliable Distance (ERD) is the winning measure in [25]. Consider the same blinking graph as in the Blink Model and let D be the shortest-path distance from A to B:

$$\mathrm{score}_{\mathrm{ERD}}(\mathrm{A,B}) = \mathrm{E}\left[D \mid D \neq \infty\right] \qquad (4)$$

Note that this is an inverse-proximity measure.

Our implementation of Adamic/Adar [1] in Section 4.2 is:

$$\mathrm{score}_{\mathrm{Adamic/Adar}}(\mathrm{A,B}) = \sum_{\mathrm{C}} \frac{n_{\mathrm{A,C}}\cdot n_{\mathrm{C,B}}}{\log d_{\mathrm{C,in}} + \log d_{\mathrm{C,out}}} \qquad (5)$$

where $n_{\mathrm{A,C}}$ is the number of A-to-C edges, $n_{\mathrm{C,B}}$ is the number of C-to-B edges, and $d_{\mathrm{C,in}}$ and $d_{\mathrm{C,out}}$ are the numbers of in-coming and out-going edges of node C.

1.3 Basic arithmetic

For clarity of presentation and without loss of generality[1], this section assumes all node weights being 1.

Exact evaluation of the Blink Model can be as follows. Enumerate all subgraphs of G, each of which is a state of the blinking graph and has a probability that is equal to the product of $w_E(e)$ for edges e that exist and $1-w_E(e)$ for edges e that do not. (2) is the sum of probabilities of subgraphs where a path exists from A to B, and (1) gets calculated accordingly. This has impractical complexity.

Monte Carlo evaluation of the Blink Model measure can be as follows. Each sample traverses the subgraph reachable from A in one instance of the blinking graph. (2) is approximated by the fraction of samples that reach B, and (1) gets approximated accordingly. This can be expensive. In Table 2, we demonstrate that at least tens of thousands of samples are needed to reliably discern different pairs of nodes. Yet in Section 4.2 for a graph that represents Wikipedia citation network, practical run time allows only 100 samples.

If two edges $e_1 = (\mathrm{X},\mathrm{Y})$ and $e_2 = (\mathrm{Y},\mathrm{Z})$ are the only edges to/from node Y, they can be replaced by a single edge from X to Z with weight $w_E(e_1)\cdot w_E(e_2)$, without altering the Blink Model measure for any pair of nodes.

Two parallel edges e_1 and e_2 can be replaced by a single edge with weight $1-(1-w_E(e_1))\cdot(1-w_E(e_2))$, without altering the Blink Model measure for any pair of nodes. x parallel edges with weight w is equivalent to a single edge with weight $1-(1-w)^x$. In other words, an edge with weight $1-(1-w)^x$ is x times as strong as an edge with weight w.

2. THE MEASURE

2.1 Properties

Let us begin with two properties of (1), *additivity* and *monotonicity*, which are important in the coming sections.

Additivity. Let $G_1 = \langle V_1, E_1\rangle$ and $G_2 = \langle V_2, E_2\rangle$ be two graphs such that $V_1 \cap V_2 = \{\mathrm{A},\mathrm{B}\}$, $E_1 \cap E_2 = \emptyset$. Let $G_3 = \langle V_1 \cup V_2, E_1 \cup E_2\rangle$ be a combined graph that has the same node and edge weights as in G_1 and G_2. Let $s_{G_1}(\mathrm{A},\mathrm{B})$, $s_{G_2}(\mathrm{A},\mathrm{B})$ and $s_{G_3}(\mathrm{A},\mathrm{B})$ be the measure value (1) in these three graphs respectively. Then this condition holds:

$$s_{G_3}(\mathrm{A},\mathrm{B}) = s_{G_1}(\mathrm{A},\mathrm{B}) + s_{G_2}(\mathrm{A},\mathrm{B}) \qquad (6)$$

Among competitors defined in Section 1.2, Adamic/Adar and EC have the same additivity property. Katz does not have this property, as in general (3) in G_3 is more than the sum of that in G_1 and G_2, and may even be divergent.

The additivity property of the proposed measure is consistent with human intuition. When multiple independent sets of evidence are combined, our perception of proximity becomes the sum of proximity values derived from each individual set. To state the same in more general terms, the proposed proximity measure (1) is proportional to the

[1]Blinking graphs with edge weights alone, setting all node weights to 1, are equally expressive. A node weight can be expressed as an edge weight by splitting a node into two nodes, one being sink of in-coming edges, one being source of out-going edges, and adding an auxiliary edge between the two, with edge weight equal to the original node weight [3].

amount of evidence, which is why we choose it over (2). This additivity property is also crucial to the development of the approximation algorithm in Section 3.

Monotonicity. Let $G_1 = \langle V_1, E_1 \rangle$ and $G_2 = \langle V_2, E_2 \rangle$ be two graphs such that $V_1 \subseteq V_2$, $E_1 \subseteq E_2$, and that their weights satisfy that $w_{V_1}(\mathrm{X}) \leq w_{V_2}(\mathrm{X}), \forall \mathrm{X} \in V_1$ and that $w_{E_1}(e) \leq w_{E_2}(e), \forall e \in E_1$. Let $s_{G_1}(\mathrm{A,B})$ and $s_{G_2}(\mathrm{A,B})$ be the measure value (1) in these two graphs respectively. Then the following condition holds.

$$s_{G_1}(\mathrm{A,B}) \leq s_{G_2}(\mathrm{A,B}), \forall \mathrm{A,B} \in V_1 \tag{7}$$

In plain language, if an edge is added to a graph or if a node/edge weight is increased in a graph, then the proposed measure (1) will not decrease for any pair of nodes. This again is consistent with human intuition.

Among competitors defined in Section 1.2, Katz and EC have the same monotonicity property, assuming that the additional edges or added weights do not cause (3) to diverge. In Adamic/Adar's (5), if the denominator is viewed as reciprocal of $w_V(\mathrm{C})$ which implies a specific choice[2] of w_V, then it also satisfies the monotonicity property. Note that (inverse) ERD is not monotonic because additional edges may form a new long path from A to B and hence increase (4).

2.2 Generalizations

The measure (1) is defined on a particular event, "a path exists from A to B". This definition is a pair-wise proximity measure and is useful in for example link-prediction applications. For other applications, the definition (1) can be generalized to other events in the blinking graph: e.g., for a set of nodes S_A and another set of nodes S_B,

$$\tilde{s}(S_\mathrm{A}, S_\mathrm{B}) = -\log(1 - \mathrm{P}[\text{ a path exists from any of } S_\mathrm{A} \text{ to each of } S_\mathrm{B}]) \tag{8}$$

Or for three nodes A, B and C,

$$\tilde{\tilde{s}}(\mathrm{A,B,C}) = -\log(1 - \mathrm{P}[\text{ a path exists from A to B but no path exists from A to C }]) \tag{9}$$

And there are many more possibilities. In particular, when edges are labeled to indicate types of relations [23], the choice of event can involve edge labels.

The measure (1) is a proximity measure. Another variation is a distance measure:

$$d(\mathrm{A,B}) = -\log(b(\mathrm{A,B})) \tag{10}$$

It is straightforward to verify that the above definition satisfies the triangle inequality. It also has the monotonicity property. It has an additivity property that differs from that in Section 2.1, but is defined on two graphs in series.

2.3 Graph studies

This section uses graph examples to compare the proposed proximity measure against competitors to demonstrate that it is more consistent with human intuition. Comparison against Adamic/Adar (5) or common-neighbor (replacing the sum in (5) by a sum of 1's) is straightforward since they are limited to length-2 paths, and an example is omitted.

[2]Such choice of weights is shown to be beneficial in for example social networks [1]. With Blink Model, this can easily be encoded as domain knowledge, to be described in Section 2.4. In fact similar schemes are used in Section 4.

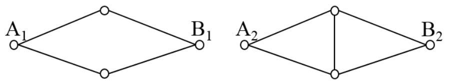


Figure 1: A pair of graph examples.

In examples in Figures 1–3, we argue that human intuition would say that node A has stronger relation to node B_2 than to B_1. A key notion is that human intuition not only prefers more and shorter paths, but also prefers structures that are mutually corroborated. If an edge or path has no corroboration, its existence in the graph may be a random coincidence and hence does not indicate strong proximity. On the flip side, proximity is strong for an aggregate structure that is impervious to edges randomly existing.

In discussing all examples, we assume uniform node weights of 1 and uniform edge weights of $w < 1$.

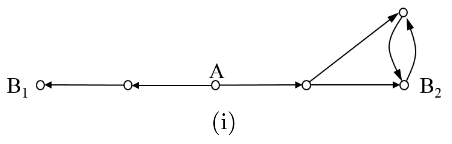
Table 1: Some proximity measures on Figure 1.

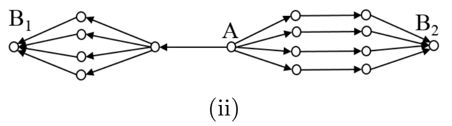
Measure	$\mathrm{A}_1,\mathrm{B}_1$	$\mathrm{A}_2,\mathrm{B}_2$
1/shortest-path	$1/(2w)$	$1/(2w)$
1/commute-time	$1/(8w)$	$1/(10w)$
EC	w	w
CFEC	w	$8w/9$

Let us begin with Figure 1 of two undirected graphs. It could be perceived that there are two random paths from A_1 to B_1, while the two length-2 paths from A_2 to B_2 are less likely to be random because the crossing edge provides mutual corroboration between them, and therefore human intuition prefers $(\mathrm{A}_2, \mathrm{B}_2)$ over $(\mathrm{A}_1, \mathrm{B}_1)$. Table 1 lists various proximity scores, where none is consistent with human intuition. Shortest-path and EC conclude that $(\mathrm{A}_1, \mathrm{B}_1)$ and $(\mathrm{A}_2, \mathrm{B}_2)$ are equally related, while CFEC [19] and commute-time [11] conclude that $(\mathrm{A}_1, \mathrm{B}_1)$ is stronger than $(\mathrm{A}_2, \mathrm{B}_2)$. In contrast, the Blink Model score is $-2 \cdot \log(1 - w^2)$ for $(\mathrm{A}_1, \mathrm{B}_1)$ and $-\log(1 - 2w^2 - 2w^3 + 5w^4 - 2w^5)$ for $(\mathrm{A}_2, \mathrm{B}_2)$, and the latter is strictly larger than the former. This shows a weakness of EC in that it sees no effect from the crossing edge in the second graph; the EC variant of CFEC [19] exacerbates this trait; another EC variant [9,29] adds a universal sink to the EC model, and it is straightforward to verify that it also ranks $(\mathrm{A}_1, \mathrm{B}_1)$ as stronger than $(\mathrm{A}_2, \mathrm{B}_2)$, and similar effects of the universal sink have been reported in [19]. A spectrum of measures was proposed in [31], where shortest-path is one end of the spectrum while commute-time is the other end; although we are unable to judge intermediate variants of [31], Table 1 suggests that both of its corner variants produce counterintuitive rankings for Figure 1.

Next let us consider Figure 2(i). There are two equal-length paths from A to B_1 and to B_2, but there are more paths going from A to B_2. So it seems there's more reason to believe that it's not a coincidence that B_2 is connected to A than B_1. PageRank scores are $\mathrm{score_{PPR}}(\mathrm{A,B_1}) = (1-\alpha)^2/2$ and $\mathrm{score_{PPR}}(\mathrm{A,B_2}) = (1-\alpha)^2/4$. In other words, PPR considers that A has greater proximity to B_1 than to B_2, and this holds true for any parameterization. In contrast, the Blink Model score (1) is higher for B_2 than B_1.

Now consider the Katz measure (3) on Figure 2(ii). It's straightforward to verify that, with any w and β values, we have $\mathrm{score_{Katz}}(\mathrm{A,B_1}) = \mathrm{score_{Katz}}(\mathrm{A,B_2})$, including when w is 1 and (3) becomes the original Katz. In other words, the (modified) Katz measure cannot discern B_1 and B_2 relative to A, because it sees no difference between the four paths to B_1 and those to B_2. In contrast, the Blink Model measure





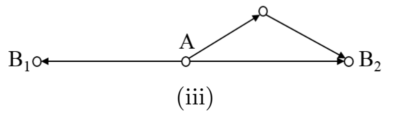


Figure 2: Graph examples for PPR, Katz and ERD.

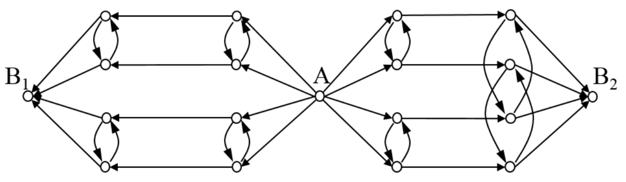


Figure 3: A graph example.

(1) is able to recognize that the edge to the left of A, which all paths to B$_1$ depend on, has no corroboration, and we have $s\left(\mathrm{A},\mathrm{B}_1\right) < s\left(\mathrm{A},\mathrm{B}_2\right)$, consistent with human intuition.

Next consider the ERD measure (4) on Figure 2(iii), and we have $\mathrm{score}_{\mathrm{ERD}}\left(\mathrm{A},\mathrm{B}_1\right) = 1$ and $\mathrm{score}_{\mathrm{ERD}}\left(\mathrm{A},\mathrm{B}_2\right) > 1$. Since ERD is an inverse proximity, the conclusion is that A has greater proximity to B$_1$ than to B$_2$, and is inconsistent with human intuition. Blink Model shows the reverse.

Next consider SimRank [14] on a three-node undirected complete graph and on a four-node undirected complete graph. It is straightforward to verify that the SimRank score, under any parameterization, is higher for a pair of nodes in the former than in the latter, and in fact the score always decreases as the size of a complete graph increases. This is contrary to our Blink Model measure and human intuition. To be fair, SimRank was designed to be a similarity measure and was not intended to be a proximity measure. This is also the likely reason that SimRank performance in [21] was reported to be mediocre.

The last example graph is Figure 3, which demonstrates the advantage of measure (1) over a class of methods. Continuing the intuition from Figure 1, on the left there exists mutual corroboration between the top pair of length-3 paths to B$_1$ and between the bottom pair of length-3 paths, but none exists between the two pairs. On the right there exists mutual corroboration among all four length-3 paths to B$_2$, and the proximity to B$_2$ is perceived as more robust to a human. This is analogous to using four strings to reinforce four poles, and a human is more likely to use a pattern similar to the right half of Figure 3 than the left. The Blink Model recognizes that B$_2$ is more connected to A than B$_1$, e.g. when w is 0.5, $s\left(\mathrm{A},\mathrm{B}_1\right) = 0.795$ and $s\left(\mathrm{A},\mathrm{B}_2\right) = 0.809$.

Consider any algorithm which operates on local storage per node: it starts with special storage in source node A, all other nodes starting equal; on each successive iteration, it updates information stored in each node based on information stored in adjacent nodes from the previous iteration. Such an algorithm can easily for example compute shortest-path distance from A, the PPR score, the Katz score, and the EC score. However, such an algorithm, even with an infinite amount of storage and an infinite number of iterations, cannot distinguish B$_1$ and B$_2$ in Figure 3; in fact, the eight nodes that are distance-2 from A are also indistinguishable. Algorithms of this type encompass a large range of methods. In particular, any matrix computation based on the adjacency matrix or variants of the adjacency matrix, which includes almost all measures that have a random-walk-based definition, falls into this category, and no possible linear algebra can determine that B$_2$ is closer than B$_1$. This is a blessing and a curse: the Blink Model can discern cases correctly, but is inherently hard to compute.

2.4 Training weights

This section addresses a practical issue of using the Blink Model in an application: how to set edge and node weights. There are many ways, and we describe a simple yet practical method to do so by training a few parameters.

Let two functions $f_E : E \to R_{>0}$ and $f_V : V \to R_{>0}$ represent domain knowledge. In applications where we have no domain knowledge beyond the topology G, we simply have f_E and f_V equal to 1 for all. In applications where we do, we assume that f_E and f_V are larger for more important or more reliable edges and nodes, and that their values exhibit linearity: two parallel edges e_1 and e_2 can be replaced by a single edge e_3 with $f_E(e_3) = f_E(e_1) + f_E(e_2)$.

Our method sets graph edge and node weights as:

$$\begin{aligned} w_E(e) &= 1 - (1 - b_1)^{f_E(e)}, \forall e \in E \\ w_V(v) &= 1 - (1 - b_2)^{f_V(v)}, \forall v \in V \end{aligned} \tag{11}$$

where $b_1, b_2 \in (0, 1)$ are two tunable parameters. It is straightforward to verify that the linearity assumption on f_E is consistent with the arithmetic in Section 1.3. Parameters b_1 and b_2 for Blink Model are similar to α for PageRank and β for Katz, and we search for best values by using training data in an application. If f_E and f_V have tunable parameters, those can be trained in the same process. Since we introduce only two parameters, the training process is straightforward and can be brute-force scan and/or greedy search.

This method is applicable to all proximity measures and is used for all in Section 4. One caveat is that certain measures work better with linear weights rather than (11):

$$\begin{aligned} w_E(e) &= b_1 \cdot f_E(e), \forall e \in E \\ w_V(v) &= b_2 \cdot f_V(v), \forall v \in V \end{aligned} \tag{12}$$

For example, we observe empirically that PPR works better with (12), which is intuitive given the linearity assumption on f_E, while Modified Katz and ERD prefer (11). Note that when b_1 and b_2 are small, (11) asymptotically becomes (12).

3. APPROXIMATION ALGORITHM

We present a deterministic algorithm from [27] that approximates (1) directly. Without loss of generality (per Section 1.3), we describe this algorithm under the conditions of all node weights being 1 and that G is a simple directed graph where parallel edges are already merged.

3.1 Overall flow

A *minimal path* from node A to node B is defined as a path from A to B without repeating nodes. For a finite graph G, there exist a finite number of minimal paths. Consider a single minimal path i from node A to node B. We define the followingas the *nominal contribution* of this path to (1).

$$s_{\text{path } i} = -\log\left(1 - \prod_{\text{edge } e \text{ on path } i} w_E(e)\right) \tag{13}$$

By the additivity property (6), if all minimal paths from A to B are mutually disjoint, we can compute (1) exactly by summing (13) over all minimal paths. Of course this is not true for general graphs where paths from A to B overlap each other and (1) is less than the sum of $s_{\text{path } i}$ values.

However, if we consider only length-1 and length-2 minimal paths, they can never share an edge with each other, and their nominal contributions can be added according to the additivity property. Further invoking the monotonicity property (7), we obtain the following inequality.

$$\sum_{\text{path } i \text{ with length 1 or 2}} s_{\text{path } i} \le s(\mathrm{A},\mathrm{B}) \le \sum_{\text{path } i} s_{\text{path } i} \tag{14}$$

Therefore, the key to approximate (1) is to quantify the contribution of minimal paths that are longer than 2.

We start the approximation by assuming a condition that the contribution of each minimal path i is quantifiable as a value $\hat{s}_{\text{path } i}$ such that $s(\mathrm{A},\mathrm{B}) = \sum_{\text{path } i} \hat{s}_{\text{path } i}$. We use G' to denote a subgraph of G and $s_{G'}(\mathrm{A},\mathrm{B})$ to denote the measure value (1) in G'.

Condition 1. *A value $\hat{s}_{path\ i}$ exists for each minimal path i from node A to node B, such that these $\hat{s}_{path\ i}$ values satisfy the following conditions:*

$$\begin{aligned} \hat{s}_{path\ i} &= s_{path\ i}, \text{ if path } i \text{ has length 1 or 2} \\ 0 \le \hat{s}_{path\ i} &\le s_{path\ i}, \text{ if path } i \text{ is longer than 2} \end{aligned} \tag{15}$$

$$\begin{aligned} s_{G'}(A,B) &\ge \sum_{path\ i\ is\ contained\ in\ G'} \hat{s}_{path\ i} \\ s_{G'}(A,B) &\le \sum_{path\ i\ overlaps\ with\ G'} \hat{s}_{path\ i} \end{aligned} \tag{16}$$

for any subgraph G'.

Our algorithm works best when Condition 1 holds, while the approximation would be coarser when it does not. We have not found any graph that breaks Condition 1, and it remains an unproven conjecture whether it holds for all graphs. We use Condition 1 in two ways. By selecting a special set of subgraphs G', we utilize (16) to approximate $\hat{s}_{\text{path } i}$ values. Then, after obtaining approximate $\hat{s}_{\text{path } i}$ values, we invoke Condition 1 for a special case of $G' = G$, where the two sums in condition (16) are identical and therefore (16) becomes two equalities. This justifies that $s(\mathrm{A},\mathrm{B}) = \sum_{\text{path } i} \hat{s}_{\text{path } i}$ which achieves our purpose.

One observation is that Condition 1 does not uniquely define $\hat{s}_{\text{path } i}$ values as there may exist two sets of $\hat{s}_{\text{path } i}$ values that both satisfy (15)(16). However, by definition they both sum up to the same end result (1), and therefore we only need to find one such set of $\hat{s}_{\text{path } i}$ values. A second observation is that the lower bound in (16) is tight for a variety of subgraphs G', while the upper bound is tight only for large subgraphs. We exploit this observation in the proposed algorithm: we will select/design a certain set of subgraphs G' where the lower bound in (16) is tight, and then use the lower bound as an equality to iteratively refine the approximated $\hat{s}_{\text{path } i}$ values.

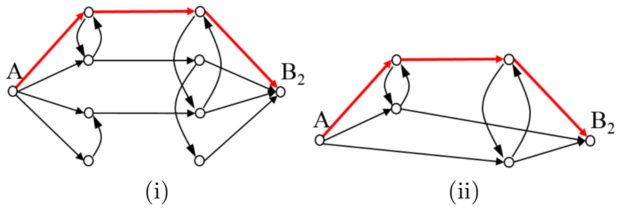


Figure 4: (i) The selected subgraph of Figure 3 for the highlighted path, and (ii) its simplified form.

The proposed algorithm initializes $\hat{s}_{\text{path } i}$ values to be the nominal values $s_{\text{path } i}$. A subgraph G'_i, to be elaborated later, is selected for each path i longer than 2, and $s_{G'_i}(\mathrm{A},\mathrm{B})$ is computed/approximated. Then during each iteration, for each path i longer than 2, $\hat{s}_{\text{path } i}$ value is updated by applying condition (16) on G'_i: we use the lower bound in (16) as an equality and convert it to the following update formula.

$$\hat{s}^{k+1}_{\text{path } i} = \frac{\hat{s}^{k}_{\text{path } i} \cdot \left(s_{G'_i}(\mathrm{A},\mathrm{B}) - \sum_{\text{path } j \in \Upsilon_i} s_{\text{path } j}\right)}{\sum_{\text{path } j \in \Xi_i} \hat{s}^{k}_{\text{path } j}} \tag{17}$$

where k is the iteration index, Ξ_i is the set of minimal paths from A to B that are contained in G'_i and that have length more than 2, and Υ_i is the set of minimal paths from A to B that are contained in G'_i and that have length of 1 or 2. After the iterations converge, we obtain an approximated (1) by summing the final $\hat{s}_{\text{path } i}$ values.

One way to interpret this algorithm is that it is an iterative solver that solves a linear system where there is one equation for each G'_i and the unknowns are $\hat{s}_{\text{path } i}$. This interpretation holds for the variation in Section 3.2, however it does not hold in Section 3.3, in which we will present an alternative interpretation.

In the next two sections, we present variations of the proposed algorithm. They differ in how they select/construct subgraph G'_i for a given path i.

3.2 High-accuracy variation

In this variation of the proposed algorithm, we select subgraph G'_i as the minimal subgraph that contains all minimal paths from A to B which overlap with path i by at least one edge. One example is illustrated in Figure 4(i), which shows the subgraph of Figure 3 for the highlighted path from A to B$_2$, and it is used in (17) to update $\hat{s}$ of the highlighted path during each iteration. Note that G'_i only needs to be identified once and $s_{G'_i}(\mathrm{A},\mathrm{B})$ only needs to be evaluated once, and the same value is reused in (17) across iterations.

A main computation in this variation is the evaluation of $s_{G'_i}(\mathrm{A},\mathrm{B})$. Since G'_i is much smaller than the whole graph G in typical applications, many techniques from the network reliability field can be applied to approximately evaluate (2) in G'_i and hence $s_{G'_i}(\mathrm{A},\mathrm{B})$. For example, it is known that (2) is invariant under certain topological transformations [3,28]. Applying such transformations, the graph in Figure 4(i) can be simplified to Figure 4(ii) without loss of accuracy. Then a Monte Carlo method can be applied on the simplified graph and approximate $s_{G'_i}(\mathrm{A},\mathrm{B})$.

3.3 Medium-accuracy variation

Instead of identifying and solving each G'_i as an actual

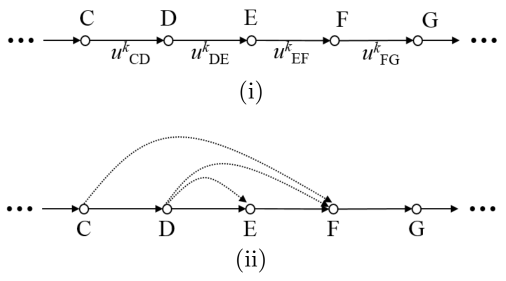


Figure 5: (i)Example of a middle section of a path. (ii)The same middle section after adding hypothetical edges.

subgraph, in this variation we construct G'_i as a hypothetical subgraph for each path i during each iteration.

We start the construction by considering the amount of sharing on each edge. In the k^{th} iteration, for edge e, define u^k_e as the sum of $\hat{s}^k_{\text{path }j}$ values over all paths j that contain e and are longer than 2. Intuitively, u^k_e quantifies usage of edge e by A-to-B minimal paths, based on current knowledge at the k^{th} iteration.

Figure 5(i) illustrates a middle section of an example path i. We use u^k_{XY} to denote u^k_e when e is an edge from node X to node Y, and w_{XY} to denote its edge weight. Without loss of generality, we assume that $u^k_{\text{FG}} > u^k_{\text{CD}} > u^k_{\text{EF}} > u^k_{\text{DE}}$.

We construct the hypothetical subgraph G'_i starting from path i itself and by adding hypothetical edges. Since $u^k_{\text{EF}} > u^k_{\text{DE}}$, there must exist one or more A-to-B path(s) that passes the edge from E to F but that does not pass the edge from D to E. A hypothetical new edge from D to E is added to approximate the effect of such path(s); furthermore, we know that the sum of $\hat{s}^k$ of these paths is equal to $u^k_{\text{EF}} - u^k_{\text{DE}}$, and we use this fact to assign the following weight.

$$w'_{\text{DE}} = 1 - (1 - w_{\text{DE}})^{\left(u^k_{\text{EF}} - u^k_{\text{DE}}\right)/u^k_{\text{DE}}} \tag{18}$$

In plain words, we assume that this hypothetical edge is $\left(u^k_{\text{EF}} - u^k_{\text{DE}}\right)/u^k_{\text{DE}}$ times as strong as original D-to-E edge.

Similarly, since $u^k_{\text{CD}} > u^k_{\text{EF}}$, there must exist one or more A-to-B path(s) that passes the edge from C to D, but that does not pass the edge from E to F, the edge from D to E, or paths represented by the hypothetical edge from D to E. A hypothetical new edge from D to F is added to approximate the effect of such path(s). Again, we know that the sum of $\hat{s}^k$ of these paths is equal to $u^k_{\text{CD}} - u^k_{\text{EF}}$, and by the same argument for (18), we assign the following edge weight.

$$w'_{\text{DF}} = 1 - \left(1 - w_{\text{EF}} \cdot \left(1 - (1 - w_{\text{DE}})^{u^k_{\text{EF}}/u^k_{\text{DE}}}\right)\right)^{\frac{u^k_{\text{CD}} - u^k_{\text{EF}}}{u^k_{\text{EF}}}} \tag{19}$$

The same rationale applies to adding the hypothetical edge from C to F, and so on.

The above construction process for G'_i processes edges on path i one by one in the order of increasing u^k_e values and adds hypothetical edges. The last step of construction is to add to G'_i the length-2 A-to-B paths that overlap with path i, since they are not visible in u^k_e values. The completed hypothetical subgraph G'_i is then used in (17) to compute $\hat{s}^{k+1}_{\text{path }i}$ for the next iteration, and the overall algorithm proceeds. Note that the denominator $\sum_{\text{path }j \in \Xi_i} \hat{s}^k_{\text{path }j}$ in (17) is simply the largest u^k_e along path i.

One distinction between this variation and Section 3.2 is that the exact evaluation of $s_{G'_i}(\text{A}, \text{B})$ has linear complexity with respect to path length. The hypothetical edges form series-parallel structures that are friendly to topological transformations [28]. Using Figure 5(ii) as an example, the hypothetical D-to-E edge and the original D-to-E edge can be merged into a single edge; then it and the E-to-F edge can be merged into a single D-to-F edge; then it and the hypothetical D-to-F edge can be merged, and so on.

Another distinction between this and Section 3.2 is that G'_i is no longer the same across iterations. As a result, the linear-system interpretation mentioned in Section 3.1 no longer holds. Instead, the following interpretation is more intuitive. Let $u^k_{\max}$ denote the largest u^k_e along path i. The calculation by (17) applies a dilution factor $\hat{s}^k_{\text{path }i}/u^k_{\max}$ on the strength of G'_i excluding length-2 paths. The more path i overlaps with other paths, the larger $u^k_{\max}$ is and the smaller the dilution factor is, and G'_i is a hypothetical subgraph that mimics a path where every edge has usage $u^k_{\max}$. By this interpretation, other variations exist that construct G'_i more crudely and hence have lower complexity and lower accuracy.

3.4 Accuracy assessment

Let us verify accuracy on Figure 3 for which we can compute exact scores. Table 2 shows three cases where all edge weights are 0.1, 0.5 and 0.9 respectively. The medium-accuracy variation of Section 3.3 is unable to distinguish B_1 and B_2 and therefore has 100% relative error in Error_Δ columns. Each Monte Carlo measurement is repeated with 100 different random number generation seeds, and the reported relative error is the average over the 100 runs. Not surprisingly, Monte Carlo favors 0.5 edge weight and has larger errors for higher or lower weights, while our algorithm is stable across the range. Table 2 suggests that the high-accuracy variation of Section 3.2 has comparable accuracy to 10K Monte Carlo samples for individual-score estimation, and is more accurate in differentiating B_1 and B_2 than one million samples for two of the three cases. It also suggests that a Monte Carlo method needs at least tens of thousands samples to reliably differentiate nodes.

4. EXPERIMENTAL RESULTS

This section compares the predictive power of our method against competitors on two temporal link prediction tasks. Data and benchmarking code are released at [26]. All blink-model runs use the variation of Section 3.3; single-thread run time is 2.9–5.0 seconds per starting node in coauthorship graphs and 5.3 seconds in the Wikipedia graph, on a Linux server with Intel E7-8837 processors at 2.67GHz.

We use the method of Section 2.4 with two scenarios: graph #1 is with no domain knowledge, f_E and f_V being 1 for all, and hence with uniform edge and node weights b_1 and b_2; graph #2 is with domain knowledge expressed in f_E and f_V. In Section 4.1, we follow the practice of [21] and scan parameters for each predictor without separating training and test sets. In Section 4.2, we separate data into a training set and a test set, and the training set is used to scan for the best parameterization, while the test set is used for evaluation. Best parameter values for each predictor are listed in Tables 3 and 4; no-effect parameters are omitted, e.g., PPR scores are invariant with respect to b_1 and b_2.

4.1 Link prediction in arXiv

This section replicates the experiments in [21]. For different areas in arXiv, given the coauthors of papers published

Table 2: Accuracy of methods on Figure 3. Error_{B_2} is relative error in $s(A, B_2)$. Error_{Δ} is relative error in $s(A, B_2) - s(A, B_1)$.

Edge weight	0.1		0.5		0.9	
	Error_{B_2}	Error_{Δ}	Error_{B_2}	Error_{Δ}	Error_{B_2}	Error_{Δ}
Our algorithm, high accuracy	0.07%	19.2%	2.40%	12.4%	2.40%	6.73%
Our algorithm, medium accuracy	8.86%	100%	17.2%	100%	12.0%	100%
Monte Carlo, 1K samples	37.7%	4.21E2	3.38%	282%	43.1%	228%
Monte Carlo, 10K samples	12.9%	1.56E2	1.07%	106%	3.82%	272%
Monte Carlo, 100K samples	3.55%	4751%	0.35%	29.2%	1.05%	70.2%
Monte Carlo, 1M samples	1.13%	1852%	0.11%	8.69%	0.33%	23.7%

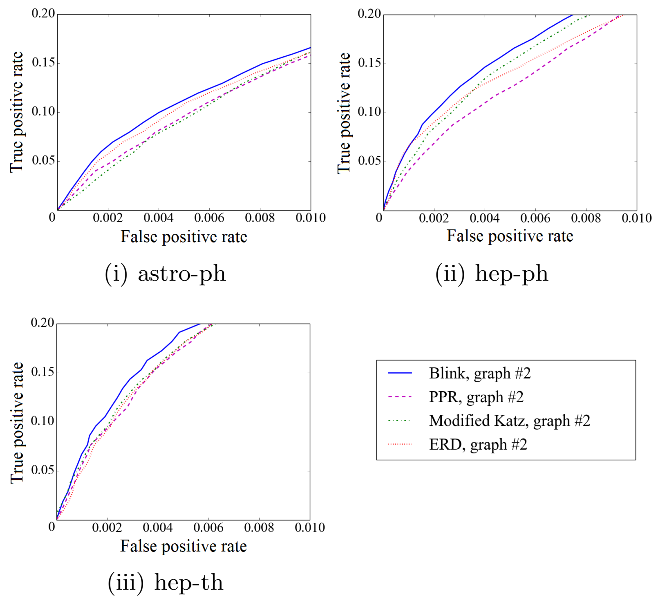


Figure 6: ROC curves on coauthorship networks.

in 1994-1996, the task is to predict new pairwise coauthorship relations formed in 1997-1999. Please refer to [26] for graph/task statistics and detailed description.

In Table 3, the oracle of [21] is the highest score for each benchmark, by all predictors including meta-approaches; note that no single predictor has such performance, and PPR and Katz on uniformly weighted graphs are dominated by the oracle. Allowing the best b_1 and b_2 per graph leads to the oracle of Blink, and for a single parameterization we get the next row. Such performance with graph #1 already puts Blink above all predictors reported in [21].

In graph #2, each paper is modeled as a node, and it connects to and from each of its authors with two directed simple edges. We provide domain knowledge through the following f_E and f_V. For an edge $e = (X, Y)$, $f_E(e) = 1/\max(1, \log_\gamma d_X)$, where d_X is the out degree of X. For a paper node, we set f_V to infinity and hence weight to 1. For an author node, $f_V(\text{author}) = 1/\max(1, \log_\gamma m_{\text{author}})$, where m_{author} is the number of coauthors of this author in the training period. γ is a tunable parameter and is scanned with other parameters and reported in Table 3. A single parameterization of Blink Model outperforms oracle of [21] by 14–35%. Please refer to [26] for detailed discussions.

Receiver-operating-characteristic (ROC) curves are plotted in Figure 6, where Blink dominates. Note that ERD #2 performs well on hep-ph for early guesses at around 5% true positive rate, but it degrades quickly after that and becomes the worst of the four by the 20% rate.

4.2 Link prediction in Wikipedia

In this experiment, the known graph is the inter-wikipage-citation graph based on Wikipedia dumps in April/May 2014, and the task is to predict new citation links added from then to March 2015. Among 93,845 pages, i.e. nodes, that have undergone meaningful edits, we randomly sample a 1000-node training set and a 1000-node test set. We further form a trimmed test set by removing links that are too easy or too difficult. Please refer to [26] for graph/task statistics and detailed data collection description.

In Table 4, mean average precision (MAP) is the accuracy score. In graph #2, we provide domain knowledge through the following f_E and f_V. For an edge from node X to node Y and that represents the i^{th} citation link on page X:

$$f_E(\text{edge}) = \frac{\delta_{Y,X}}{\max\left(1, \log_\gamma i\right) \cdot \max\left(1, \log_\gamma d_{Y,\text{in}}\right)} \qquad (20)$$

where $\delta_{Y,X}$ is 2 if edge exists from Y to X, and 1 otherwise. γ is again a tunable parameter. The above scheme gives higher weight to a citation link if it is located at an earlier location on a page, or if it points to a less-cited page, or if a returning citation exists. Our f_V function is a direct adaptation of Adamic/Adar's (5): $f_V(\text{node}) = 1/(\log d_{\text{node,in}} + \log d_{\text{node,out}})$. Blink Model with graph #2 is the best performer in Table 4. Please refer to [26] for detailed discussions.

5. CONCLUSIONS

This manuscript proposes the Blink Model graph proximity measure. We demonstrate that it matches human intuition better than others, develop an approximation algorithm, and empirically verify its predictive accuracy.

6. REFERENCES

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

[2] C. C. Aggarwal et al. Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In *KDD*, pages 201–212, 1999.

[3] M. O. Ball et al. Network reliability. *Technical Report, University of Maryland*, 1992.

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

[5] V. D. Blondel et al. A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM Review*, 46(4):647–666, 2004.

[6] B. Bollobas and O. Riordan. *Percolation*. Cambridge University Press, 2006.

[7] T. B. Brecht and C. J. Colbourn. Lower bounds on two-terminal network reliability. *Discrete Applied Mathematics*, 21(3):185–198, 1988.

Table 3: Comparison of predictor accuracies on coauthorship networks. A denotes the true positive rate when the number of predictions is equal to the number of positives. R denotes the ratio of A over that of oracle of [21].

	parameters	astro-ph		hep-ph		hep-th	
		A	R	A	R	A	R
Oracle of [21]	varying	8.55%		7.2036%		7.9407%	
Oracle of Blink, graph #1	varying	9.075%	1.061	8.3816%	1.164	8.8592%	1.116
Blink, graph #1	$b_1 = 0.5$, $b_2 = 0.4$	7.7461%	0.906	7.8025%	1.083	8.0306%	1.011
Blink, graph #2	$b_1 = 0.8$, $b_2 = 0.6$, $\gamma = 5$	10.264%	**1.200**	9.6922%	**1.345**	9.0504%	**1.140**
PPR, graph #2	$\alpha = 0.50$	8.5330%	0.998	6.7358%	0.935	7.9031%	0.995
Modified Katz, graph #2	$b_1 = 0.5$, $b_2 = 0.1$, $\beta = 0.1$, $\gamma = 5$	8.4106%	0.984	8.2292%	1.142	7.8394%	0.987
ERD, 10K samples, graph #1	$b_1 = 0.9$, $b_2 = 0.9$	8.4281%	0.986	8.1682%	1.134	7.1383%	0.899
ERD, 10K samples, graph #2	$b_1 = 0.9$, $b_2 = 0.9$, $\gamma = 4$	9.5471%	1.117	8.7473%	1.214	7.1383%	0.899

Table 4: Comparison of predictor accuracies on additions of inter-wikipage citations in Wikipedia. Each predictor uses its best parameters selected based on the training set. R denotes the ratio of MAP of a predictor over MAP of Adamic/Adar.

	parameters	training		test		trimmed test	
		MAP	R	MAP	R	MAP	R
Adamic/Adar		0.0291		0.0281		0.0163	
Blink, graph #1	$b_1 = 0.5$, $b_2 = 0.1$	0.0295	1.014	0.0263	0.937	0.0166	1.019
Blink, graph #2	$b_1 = 0.8$, $b_2 = 0.8$, $\gamma = 10$	0.0362	1.244	0.0362	**1.289**	0.0233	**1.428**
PPR, graph #1	$\alpha = 0.5$	0.0299	1.029	0.0291	1.038	0.0186	1.140
PPR, graph #2	$\alpha = 0.2$, $\gamma = 500$	0.0321	1.104	0.0309	1.100	0.0206	1.263
Modified Katz, graph #1	$\beta =$ 5E-6	0.0269	0.925	0.0241	0.860	0.0151	0.924
Modified Katz, graph #2	$b_1 = 0.8$, $b_2 = 0.8$, $\beta = 0.1$, $\gamma = 10$	0.0341	1.173	0.0328	1.170	0.0198	1.213
ERD, 100 samples, graph #1	$b_1 = 0.4$, $b_2 = 0.9$	0.0266	0.914	0.0233	0.830	0.0162	0.996
ERD, 100 samples, graph #2	$b_1 = 0.9$, $b_2 = 0.9$, $\gamma = 10$	0.0238	0.817	0.0218	0.778	0.0154	0.944

[8] P. Chebotarev and E. Shamis. Matrix-forest theorem and measuring relations in small social groups. *Autom. and Remote Control*, 58(9):1505–1514, 1997.

[9] C. Faloutsos et al. Fast discovery of connection subgraphs. In *KDD*, pages 118–127, 2004.

[10] G. Fishman. Comparison of four monte carlo methods for estimating the probability of s-t connectedness. *IEEE Trans. Reliability*, 35(2):145–155, 1986.

[11] F. Fouss et al. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Trans. Knowl. and Data Eng.*, 19(3):355–369, 2007.

[12] E. N. Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

[13] P. Jaccard. The distribution of the flora in the alpine zone. *New Phytologist*, 11(2):37–50, 1912.

[14] G. Jeh and J. Widom. Simrank: a measure of structural-context similarity. In *KDD*, pages 538–543, 2002.

[15] R. Jin et al. Distance-constraint reachability computation in uncertain graphs. *VLDB*, 4(9):551–562, 2011.

[16] D. R. Karger. A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem. *SIAM Review*, 43(3):499–522, 2001.

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

[18] D. Kempe et al. Maximizing the spread of influence in a social network. In *KDD*, pages 137–146, 2003.

[19] Y. Koren et al. Measuring and extracting proximity in networks. In *KDD*, pages 245–255, 2006.

[20] E. Leicht et al. Vertex similarity in networks. *Physical Review E*, 73(2):026120, 2006.

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

[22] L. Liu et al. Reliable clustering on uncertain graphs. In *ICDM*, pages 459–468, 2012.

[23] M. Nickel et al. A review of relational machine learning for knowledge graphs: From multi-relational link prediction to automated knowledge graph construction. *Center for Brains, Minds and Machines Memo No. 028*, 2015.

[24] L. Page et al. The pagerank citation ranking: bringing order to the web. *Technical Report, Stanford University*, 1999.

[25] M. Potamias et al. K-nearest neighbors in uncertain graphs. *VLDB*, 3(1):997–1008, 2010.

[26] H. Qian. Link prediction benchmarks, 2016. [Available at http://researcher.watson.ibm.com/group/6672].

[27] H. Qian and H. Wan. Ranking related objects using blink model based relation strength determinations. *U.S. Patent Application 14/791789*, 2015.

[28] A. Satyanarayana and R. K. Wood. A linear-time algorithm for computing k-terminal reliability in series-parallel networks. *SIAM Journal on Computing*, 14(4):818–832, 1985.

[29] H. Tong et al. Fast direction-aware proximity for graph mining. In *KDD*, pages 747–756, 2007.

[30] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[31] L. Yen et al. A family of dissimilarity measures between nodes generalizing both the shortest-path and the commute-time distances. In *KDD*, pages 785–793, 2008.