Boosting SimRank with Semantics

Tova Milo, Amit Somech and Brit Youngmann
Tel Aviv University
{milo,amitsome,brity}@post.tau.ac.il

ABSTRACT

The problem of estimating the similarity of a pair of nodes in an information network draws extensive interest in numerous fields, e.g., social networks and recommender systems. In this work we revisit SimRank, a popular and well studied similarity measure for information networks, that quantifies the similarity of two nodes based on the similarity of their neighbors. SimRank's popularity stems from its simple, declarative definition and its efficient, scalable computation. However, despite its wide adaptation, it has been observed that for many applications SimRank may yield inaccurate similarity estimations, due to the fact that it focuses on the *network structure* and ignores the *semantics* conveyed in the node/edge labels. Therefore, the question that we ask is *can SimRank be enriched with semantics while preserving its advantages?*

We answer the question positively and present SemSim, a modular variant of SimRank that allows to inject into the computation any semantic similarly measure, which satisfies three natural conditions. The probabilistic framework that we develop for SemSim is anchored in a careful modification of SimRank's underlying random surfer model. It employs Importance Sampling along with a novel pruning technique, based on unique properties of SemSim. Our framework yields execution times essentially on par with the (semantic-less) SimRank, while maintaining negligible error rate, and facilitates direct adaptation of existing SimRank optimizations. Our experiments demonstrate the robustness of SemSim, even compared to task-dedicated measures.

1 INTRODUCTION

Estimating node similarity in information networks is the cornerstone of many applications, e.g., retrieving similar users in social networks, and a fundamental component in numerous network analysis algorithms, such as link prediction and clustering.

In this work we consider SimRank [13], a well-studied similarity measure for information networks. The intuition behind SimRank is that similar objects are referenced by similar objects, and thus it quantifies node similarity based on the compound similarity of their neighbors. SimRank's popularity stems from its simple declarative definition and its efficient computation, incorporating a broad range of optimizations [15, 39]. However, despite its wide adaptation, it has been observed that for many applications SimRank may yield inaccurate estimations [37, 40], as it focuses solely on the *network structure* and ignores the *semantic information* conveyed in the node/edge labels. Thus, the question we address is the following:

Can SimRank be enriched with semantics while preserving its intuitive, declarative definition and efficient computation?

We answer the question positively, and present SemSim, a refined variant of SimRank, that weights nodes' structural similarity with their semantic similarity and edge weights, yielding an

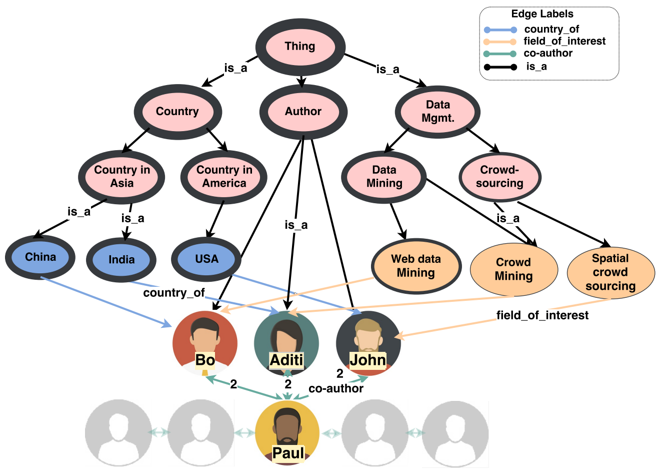


Figure 1: Example information network.

effective, comprehensive measure. SemSim's probabilistic framework, anchored in a careful modification of SimRank's underlying random surfer model, together with dedicated optimizations, allows for execution times essentially on par with the semantic-less SimRank, while maintaining similar negligible error rates. Direct adaptation of existing SimRank optimizations is also enabled.

We demonstrate the problem that we tackle with an illustrative example.

Example 1.1. The simple information network depicted in Figure 1 represents a bibliographic database. It includes nodes describing authors, countries and research fields, with edges linking authors to their co-authors, country of origin and fields of interest. A semantic taxonomy is also reflected in this network (pink nodes) where entities are linked to their hypernyms, as indicated by the "is-a" edges. Edge weights reflect the strength of the relations (for conciseness, some weights are omitted but should be assumed to have an identical arbitrary default value). To visually represent the prevalence of a concept in the dataset, we use the width of the borders surrounding the nodes (an explanation of this quantification is provided in the sequel).

We wish to determine which of the authors, Bo or John, is more similar to Aditi. Observe that: (1) all three collaborated with Paul twice (as indicated by the edge weights); (2) their origin countries are all highly prevalent (as indicated by the borders' width), compared to the authors' fields of interest, thus the latter is more *informative* and should have a greater effect on the similarity[1]; (3) Crowdsourcing, the common field of John and Aditi, is more particular (less prevalent) than Data Mining, the field shared by Bo and Aditi. Hence, intuitively, Crowd Mining is semantically closer to Spatial Crowdsourcing than to Web Data Mining. Consequently, John is more similar to Aditi than Bo, even though they reside in different continents. Note, however, that ignoring semantics and considering the *network structure* alone (even including edge weights), Bo and Aditi seem more similar,

[1]Following standard argumentation, an estimation of similarity increases more drastically when indicated by a less frequent event [32].

and indeed, SimRank (like other measures [37, 43]) erroneously derive a higher similarity score for them.

Several refinements for SimRank have been proposed in the literature (see Related Work). For instance, SimRank++ [2] is a variant of SimRank that also considers edge weights, yet semantics is ignored, and, importantly, scalability is not addressed. Other works (e.g. [37]) partially account for semantics by considering only meaningful *meta-paths* (i.e. paths of a specific label patterns) between objects. But as can be seen in the above example (and in our experiments) this does not always suffice to accurately differentiate objects. Alternatively, several semantic measures have been proposed [23, 32], but they typically gauge the similarity based on ontological information and the *Information Content* (IC) of nodes, while the rest of the network structure is ignored. In an attempt to fully account for both structure and semantics, recent works abandon SimRank and rely instead on representation learning, using techniques such as node embedding [4, 30]. While this approach often outperforms a naive combination of structural and semantic similarity measures, a key drawback is that results are hard to explain and interpret, as is often the case with machine learning. Interestingly, we show that our SimRank variant not only retains its intuitive, declarative flavor but also yields more accurate estimations compared to these works.

Next, we provide a brief overview of our results.

The SemSim *similarity measure.* We refine SimRank by weighting nodes' neighbors similarity with their semantic similarity and edge weights. Our definition is modular and allows to inject into the computation any semantic measure, as long as it satisfies three intuitive conditions that are typically satisfied by existing measures. We present SemSim iterative formulation, analogous to SimRank iterative formulation [13]. We prove that SemSim's solution always exists (as was shown for SimRank), and show that its iterative formula converges to its fix-point at least as fast as SimRank, and possibly faster, due to an additional semantic factor (Section 2).

Random-surfer model. SimRank's underlying random-surfer model serves as the basis for many of its optimizations. We establish a corresponding model for SemSim. First, we define the notion of a *Semantic-Aware Random Walk* (SARW), which refines the traditional random walk definition, and prove that SemSim can be computed using SARWs. This interpretation considers the *node-pair graph* G^2, in which a node represents an ordered pair of nodes from the original graph G. Interestingly, we prove that given a threshold s.t. only similarity scores above it are of interest, the semantics can effectively be used to reduce the size of G^2, with the computation of SemSim over the reduced graph yielding the same results as those computed via the full graph G^2. Our experiments demonstrate that a significant reduction of up to three orders of magnitude is achieved in multiple datasets (Section 3).

Approximated similarity scores. Exact computation may still be expensive for large graphs, despite the speed-up gained by the graph reduction. For SimRank, the excessive size of G^2 motivated a battery of optimizations based on a Monte-Carlo (MC) procedure [15, 34, 39]. SimRank basic MC framework returns an approximated SimRank score in $O(n_w \cdot t)$ time, where n_w is the number of walks sampled from each node and t is a bound on their length. To efficiently approximate SemSim we develop an analogous MC framework, thereby enabling a direct application of SimRank optimizations. First, we show that a naïve solution of simply replacing SimRank's underlying uniform distribution with the semantic-aware distribution leads to a quadratic increase of the sample size. To overcome this, we employ *Importance Sampling*, and devise an unbiased estimator for SemSim, which avoids this increase and returns the estimated SemSim score in average time of $O(n_w \cdot t \cdot d^2)$, where d is the average in-degree in the graph G. To further reduce computation cost, we devise a dedicated pruning technique that avoids the computations of unpromising node-pairs and irrelevant (low probability) walks, at the cost of a controlled additive error to the approximated scores. While the worst-case time complexity remains the same, our experiments show pruning to be extremely effective in practice, yielding running times on par with SimRank (Section 4).

Experimental study. We conduct an experimental study over real data, demonstrating the effectiveness of SemSim in multiple practical scenarios. Our results demonstrate the robust quality of SemSim, even compared to task-dedicated measures. The results further exhibit the efficiency and accuracy of our framework and its ability to boost SimRank with semantics while preserving its performance (Section 5).

Finally, related work and conclusions are presented in sections 6 and 7, respectively. For space constraints, proofs are deferred to a technical report [27].

2 PRELIMINARIES

We first explain the data model used in our setting, then present our novel measure, SemSim.

2.1 Data Model

Following [36], we refer to the objects graph as a *Heterogeneous Information Network* (HIN), a flexible graph model that can capture and integrate various types of data. Let $\mathcal{V}$ be the domain of vertices, $\mathcal{L}$ the domain of labels, and $\mathcal{R}^+$ the domain of real positive numbers > 0.

Definition 2.1 (Heterogeneous Information Network). A HIN is a directed weighted graph $G = (V, E, \phi, \psi, W)$, where: $V \subseteq \mathcal{V}$ is a finite set of vertices; $E \subseteq V \times V$ is a set of edges; $\phi : V \longrightarrow \mathcal{L}$ and $\psi : E \longrightarrow \mathcal{L}$ are vertex and edge labeling functions, resp., and $W : E \longrightarrow \mathcal{R}^+$ is an edge weight function.

The edge weight function W associates each edge with a real positive number indicating the strength of the relation. When no knowledge about this strength is available, the weight is set to a default value. For example, in Figure 1, weights are available only for edges with the label *co-author*, where the weights reflect the number of collaborations between authors. Since no information about other weights is available, all other weights were set to 1. For a node v, we denote by $I(v), O(v)$ the set of in and out neighbors of v, resp. An individual in-neighbor is denoted as $I_i(v)$, for $1 \leq i \leq |I(v)|$, if $I(v) \neq \emptyset$ ($O_i(v)$, resp). Throughout the paper we use the variables u, v, u', v' to denote nodes in V. Here we consider directed graphs, but stress that all results can be adapted to the undirected model with minor modifications.

In many cases, the HIN is composed of two subgraphs that are aligned together: one consists of individual objects and their relations, e.g. authors/countries, and their collaboration/residence relationships. The second, ontological-style subgraph is comprises of semantic categories and relationships, e.g. the pink nodes and their "is-a" relations. Objects of the former can be connected to their corresponding categories. For example, in Figure 1 all author nodes are connected to the category *Author*. When semantic information is not included, one can enrich the graph by aligning it with publicly available ontology [5, 26] by applying existing entity alignment tools [28]. Ontologies typically contain a hierarchical taxonomy of concepts, e.g., that *USA* "is-a" a *Country in America* and *Country in America* "is-a" *Country*. Such taxonomies are often leveraged to define semantic similarity measures.

2.2 Similarity Notions

As described above, our semantic-rich graph model contains various types of linked entities, as well as additional knowledge that is captured by the edge weights. Next, we devise a refined version of SimRank [13] that effectively considers all information. We start with a short background on SimRank, then provide the formal definition for SemSim.

SimRank follows the intuitive assumption that: "two nodes are similar if they are related to similar nodes". Formally, given two nodes $u, v \in V$, their SimRank score is defined as follows. If $u = v$ then $simrank(u, v) = 1$, else: $simrank(u, v)$ is given by the following recursive formula *without the red colored parts.*

$$sim(u, v) = \quad (1)$$

$$\frac{sem(u,v) \cdot c}{N_{u,v}} \sum_{i}^{|I(u)|} \sum_{j}^{|I(v)|} sim(I_i(u), I_j(v)) \cdot W(I_i(u), u) \cdot W(I_j(v), v)$$

where c is a decay factor in $(0, 1)$, $N_{u,v} = |I(u)| \cdot |I(v)|$ and $sim(\cdot, \cdot)$ is the SimRank score of the neighboring pair-nodes. If $I(u)$ or $I(v)$ are $\emptyset$, then the score is defined to be zero.

In SimRank, the assumed graph model is an unweighted homogeneous graph, where all edges and nodes belong to a single type, thus it ignores the labels' semantics and edge weights. We enrich SimRank by weighting, at each step of the computation, the neighbors' similarity with the edge weights and the nodes' semantic similarity. Formally, given a semantic similarity measure $sem(\cdot, \cdot)$, the red parts indicate our refinements to SimRank standard formula: (i) an additional semantic factor is added; (ii) the edge weights are taken into consideration. Correspondingly, the normalization factor is set to:

$$N_{u,v} = \sum_{i}^{|I(u)|} \sum_{j}^{|I(v)|} W(I_i(u), u) \cdot W(I_j(v), v) \cdot sem(I_i(u), I_j(v))$$

where $sim(\cdot, \cdot)$ in the refined formula denotes the refined similarity of the neighbors. Here too, if $I(u)$ or $I(v)$ are the $\emptyset$, then the similarity score is defined to be zero.

Note that according to our definition of similarity, the semantic similarity of the neighboring pairs of nodes appears as well (as the definition is recursive), and therefore, the similarity of two nodes u, v is, in fact, proportional to the semantic similarity of their neighbors.

Importantly, our definition of SemSim considers all neighbor-pairs. An alternative could be to take edge labels into consideration and restrict attention to neighbor-pairs that are pointed by edges having the same label. However, while such formulation requires only minimal technical changes and all our results remain unchanged, our experiments showed it to be less accurate, as this definition may overlook possibly important relations among the objects. Moreover, both definitions yield essentially the same running times and we thus omit this restriction.

Semantic Similarity. Multiple semantic measures have been proposed in the literature [16, 20]. In general, any similarly function $sem(\cdot, \cdot)$ can be employed in SemSim, as long as it satisfies the following constraints. For all $u, v \in V$:

(1) *Symmetry.* $sem(u, v) = sem(v, u)$.
(2) *Maximum self similarity.* $sem(u, u) = 1$.
(3) *Fixed value range.* $sem(u, v) \in (0, 1]$.

Those requirements are used to prove the soundness of SemSim (Theorem 2.3). The first two are typically satisfied by common semantic measures (e.g., [16, 23, 32]). For the third constraint, scores can be normalized into a $[0 + \epsilon, 1]$ range, for a small $\epsilon > 0$ value [29].

We next briefly overview a simple and effective semantic measure that we have used in our experiments (see Section 6 for a discussion on alternatives). Lin [23] is an *Information Content* (IC)-based measure that is defined over concept taxonomies. The IC of a node v is quantify as the negative of its log likelihood: $IC(v) = -\log(P[v])$, where $P[v]$ denotes the frequency of v. I.e., the more prevalent a concept is, the lower its IC value. Intuitively, the similarity between concepts measures the ratio of the amount of information needed to state their commonality to the information needed to describe them. Given two nodes u and v in a taxonomy, their Lin score is defined as:

$$Lin(u, v) = \frac{2 \cdot IC(LCA(u, v))}{IC(u) + IC(v)}$$

where $LCA(u, v)$ is the lowest common ancestor of u and v in the taxonomy.

Note that Lin satisfies the constraints, only if the IC values are in $(0, 1]$ (proof omitted). To estimate the IC of a concept, we adapted the Seco formula [33] in our implementation, providing a simple linear-time (in the size of the taxonomy) algorithm and extended it to our setting. This adaptation ensures the IC values lie within $(0, 1]$ (see [27] for more details).

Entity	IC value
Thing	0.001
Author, Country	0.01
Country in Asia, Country in America	0.015
China, India, US	0.02
Data Management	0.2
Data Mining	0.3
Crowdsourcing	0.85
Web data mining	0.7
Crowd Mining, Spatial Crowdsourcing	0.9
Bo, John, Aditit, Paul	1.0

Table 1: IC values for Figure 1 entities.

We next provide the full computation of SimRank and SemSim for the example introduced in the Introduction (Example 1.1), while using Lin as the integrated semantic measure.

Example 2.2. We computed the IC values (depicted in Table 1) on the same domain ontology used for the AMiner dataset (which includes a taxonomy of CS terms as well as a geographic taxonomy, see experimental results), and set absent edge weights to 1. For both SimRank and SemSim, we set the decay factor c to 0.8 and the number of iterations k was set to 3.

We first review the relevant Lin scores: since all author-nodes are leafs in the taxonomy, their corresponding IC values are all 1, thus *Lin(Bo,Aditi)* = *Lin(John,Aditi)*= 0.01 (which also serves as the upper bound on their SemSim scores). Using the IC values above, we get: *Lin(Spatial Crowdsourcing,Crowd Mining)*= 0.94 and *Lin(Web Data Mining,Crowd Mining)*= 0.37. Next, we briefly overview SemSim and SimRank computation. At the first iteration, for both measures, $R_0 = 0$ for all authors pairs. Iteratively, at the next step, since all three authors share two common neighbors, *Author* and *Paul*, yet the common field-of-interest of Aditi and John is more semantically similar than the common field of Aditi and Bo, we get for SemSim that: R_1*(John,Aditi)* = 0.0073, while R_1*(Bo,Aditi)*= 0.066. Note that in this step the semantic similarity of common neighbors propagates into the computation. On the other hand, according to SimRank, in this step both pairs similarity scores are equal to 0.1. At the last step, according to SemSim R_2*(John,Aditi)* = 0.0076, while R_2*(Bo,Aditi)*= 0.0073, thus, SemSim obtains the desire result that while all authors are fairly similar, John's similarity to Aditi is a bit greater than Bo's. In contrast, according to SimRank, R_2*(John,Aditi)* = 0.12, while R_2*(Bo,Aditi)*= 0.16. These results are due to the fact that both Aditi and Bo reside in the same continent.

We also computed the SimRank scores solely over the collaboration network (i.e., ignoring the semantic relations). Not surprisingly, since the resulted network is symmetric, the obtained similarity scores for both pairs were exactly the same.

2.3 Basic Properties of SemSim

We next show a few of SemSim's properties which will then be used to present a naïve algorithm for computing SemSim, that serves as a baseline which we improve in the following sections.

Following SimRank's iterative form [13], a solution to Equation (1) can be reached by iterating to a fix-point. For the k-th iteration, an iterative function $R_k(u, v)$ denotes the similarity score of u and v in the k-th iteration. Initially, $R_0(u, v)$ is defined as 0 if $u \neq v$ and 1 otherwise. Iteratively, $R_{k+1}(u, v)$ is computed from $R_k(\cdot, \cdot)$ as follows:

$$R_0(u,v) = \begin{cases} 0, u \neq v \\ 1, u = v \end{cases} \tag{2}$$

$$R_{k+1}(u,v) = \tag{3}$$

$$\frac{sem(u,v)\cdot c}{N_{u,v}} \sum_{i}^{|I(u)|} \sum_{j}^{|I(v)|} R_k(I_i(u), I_j(v)) \cdot W(I_i(u), u) \cdot W(I_j(v), v)$$

We can prove that the iterative SemSim form has the following properties:

THEOREM 2.3. *$\forall u, v \in V$ and for every $0 \leq k \in \mathbb{N}$:*
(1) Symmetry. $R_k(u, v) = R_k(v, u)$.
(2) Maximum self similarity.$R_k(u, u) = 1$.
(3) Monotonicity. $0 \leq R_k(u, v) \leq R_{k+1}(u, v) \leq 1$.
(4) Existence. The solution always exists.
$0 \leq c < min(argmin_{N_{u,v}}(N_{u,v}), 1)$*, the solution is unique.*

First, note that the decay factor's upper bound can be found in average time of $O(n^2 \cdot d^2)$, where d is the average in-degree in the graph, by simply iterating over all node-pairs. Second, we observe that the uniqueness property here is a weaker version than the one that was proven for SimRank, where the solution is unique for every $0 \leq c < 1$. Yet, our experiments show that for real-life networks, the upper bound is high enough to comfortably accommodate typical c values chosen for SimRank (e.g., 0.6, as used in [24, 39]).

We can also show (following similar proof for SimRank [46]) that not only the scores are monotone (i.e, $R_k(u, v) \leq R_{k+1}(u, v)$), their differences in consecutive iterations are bounded.

PROPOSITION 2.4. *For every $u, v \in V$ and $k > 0$: $0 \leq R_{k+1}(u, v) - R_k(u, v) \leq sem(u, v) \cdot c^{k+1}$*

This suggests that the iterative form of SemSim converges as fast as SimRank (where the convergence was shown to be c^{k+1} [46]), and possibly faster due to the additional semantic factor. Another useful property is that $sem(\cdot, \cdot)$, the semantic similarity of two nodes, provides a natural upper bound on their SemSim score. This property is highly effective since, as we will show, it can be used to prune un-promising node-pairs.

PROPOSITION 2.5. *For every two nodes $u, v \in V$: $sim(u, v) \leq sem(u, v)$.*

To conclude, Theorem 2.3 provides a simple algorithm for computing SemSim, that computes its iterative form to its fix-point (or up to a required precision bound). We assume that the computation of a single-pair semantic similarity score can be done in constant time (possibly after pre-processing), without materializing the $n \times n$ matrix of scores. Indeed, this is the case for numerous semantic measures [16, 32], Lin's measure included. Given this, the complexity of the iterative algorithm is equivalent to SimRank's complexity [13]: The time complexity is $O(k \cdot d^2 \cdot n^2)$, where $n = |V|$, d is the average in-degree in G and k is the number of iterations. The worst case complexity for a given k is $O(n^4)$.

3 RANDOM SURFER-PAIRS MODEL

The iterative algorithm provided in the previous section has two main disadvantages: (i) it computes all pair-wise scores, even if one is interested only in a single-pair, and (ii) its complexity is prohibitive for large graphs. To address these issues, we provide an alternative interpretation to SemSim, based on the *random surfer model* for SimRank, then, explain how SemSim can be computed efficiently. In essence, we show that with careful adjustments, an analogous random surfer model can be establish for SemSim. The key challenge is to incorporate semantics. We show that SemSim measures how soon two random surfers are expected to meet, if they start in two nodes and randomly walk on the graph backward, while being aware of both edge weights and semantics. We define *Semantic-Aware Random Walks* (SARW), then prove that SemSim can be computed using them. Interestingly, we will see that semantics can be leveraged to speed up the computation.

3.1 Semantic-Aware Random Walks (SARW)

Following [13], we use the definition of a *node-pair graph* G^2, in which each node represents an ordered pair of nodes from G. An edge $e = ((u, u'), (v, v')) \in G^2$ iff both (u, v) and (u', v') are $\in G$. We extend the definition with an assignment of weights: The weight of an edge $e = ((u, u'), (v, v'))$ is defined as: $W_{G^2}(e) := W(u, v) \cdot W(u', v')$. For simplicity, we use the notation of $W(e)$ to indicate an weight in both G and G^2, when the context is clear.

Let us assume that all edges in G have been reversed. For example, Figures 2a and 2b display a graph G and all out-edges from *(A,B)* (after reversal). For simplicity, all edge weights are set to 1. We call a node $(u, v) \in V^2$ a *singleton node* if $u = v$. In SimRank, a surfer chooses the next node uniformly at random out of all out-neighbors of the current node. To incorporate semantics and weights, we devise the following distribution.

Definition 3.1 (Semantic-Aware Probability Distribution). The probability a random surfer traveling G^2 in a current node (u, u') would next move to its out-neighbor (v, v') is:

$$P[(u,u') \to (v,v')] := \frac{W((u,u'),(v,v')) \cdot sem(v,v')}{\sum_{i=1}^{|O((u,u'))|} W((u,u'), O_i(u,u')) \cdot sem(O_i(u,u'))}$$

Using the distribution above, we define SARWs as follows. A walk in G^2 represents a pair of walks in G. Let $w = \langle w_1, ..., w_k \rangle$ denote a walk in G^2, where $w_1, ..., w_k \in V^2$, and $l(w) = |w|$. The walk w has the probability $P[w]$ of traveling within it, where $P[w] := \prod_{i=1}^{k-1} P[w_i \to w_{i+1}]$.

Importantly, this distribution defines a positive probability to *all* paths in G^2. However, as the choice of the next step relies on the semantic similarity of node-pairs, pairs of higher semantic similarity are preferred over pairs of low similarity (typically, pairs whose nodes belong to different categories). Namely, paths that share the same edge label in each step, are likely to get higher probabilities. Nonetheless, even paths that do not share the same labels are considered, as they may also provide relevant information[2]. We provide here an illustrative example for a computation of SARWs.

Example 3.2. Consider again Figure 2b. Observe that author A's current country is *Canada*, and author B's origin country is the *USA*. Noticeably, even though the two edges do not share the same label, this information may be important when assessing similarity. According to our definition we get that since the entities *Canada* and *USA* are semantically similar ($Lin(Canada, USA) = 0.8$), the probability that a random surfer in the node *(A,B)* will move next to its neighbor *(Canada, USA)* is:

$$P[(A,B)\rightarrow(Canada, USA)] = \frac{0.8}{0.8+0.2+0.2+1.0} = 0.36$$

On the other hand, as the two entities *Author* and the *USA* are not semantically similar ($Lin(Author, USA) = 0.2$), the corresponding probability is lower:

$$P[(A,B)\rightarrow(Author, USA)] = \frac{0.2}{0.8+0.2+0.2+1.0} = 0.09$$

The SimRank score of a node $(u,v) \in V^2$ can be computed using all walks from it leading to a singleton node in G^2. Analogously, we prove that SemSim can be computed using all semantic-aware walks from (u,v) leading to singleton nodes in G^2. Let $W = \{(u,v) \rightsquigarrow (x,x)\}$ be the set of all walks in G^2 form (u,v) to any singleton node (x,x). If no such paths exist, then $W = \emptyset$. By definition, (x,x) is the only singleton node in w (after the first meeting, the two surfers halt). Let: $s'(u,v) = sem(u,v)\sum_{w\in W} P[w]\cdot c^{l(w)}$. Theorem 3.3 provides an alternative way to compute the SemSim score of a single pair.

Theorem 3.3. *$\forall u,v$, given c which ensures the uniqueness of $sim(\cdot,\cdot)$: $s'(u,v) = sim(u,v)$*

Using our refined model, one may compute (single pair or all pairs) SemSim scores over G^2. However, for large graphs, its size may be too large. We next explain how the semantics can be effectively employed to reduce the size of G^2.

3.2 Reducing the Size of G^2

In many practical applications one is interested only in node-pairs whose similarity scores are above a given threshold. Semantics provides an efficient tool to prune G^2 in such situations. Intuitively, Prop. 2.5 provides a semantic-based upper bound on the similarity scores, which can be used to avoid materializing un-promising node-pairs. We devise a reduced version of G^2 on which the computation of SemSim (for node-pairs with similarity scores are above a given threshold) yields the same result as that computed via the full graph G^2. Indeed, our experimental results demonstrate a significant reduction in the size of G^2.

Given a threshold $0 < \theta < 1$, we define the graph G^2_θ, which includes only pairs s.t. their semantic scores are $> \theta$. However, simply omitting nodes from G^2 directly affects the similarity scores, thus, may lead to inaccurate scores. We therefore incorporate omitted paths by refining the edge weights and possibly adding new edges. Intuitively, each omitted path is replaced by a corresponding edge, whose weight reflects its probability. If such an edge already exists in G^2, the omitted edge's weight is added to the existing edge weight. Moreover, the weight of omitted path is further weighted by the decay factor c, to ensure the similarity scores would not be affected. Last, to ensure that the probability of choosing a neighbor remains the same as in the original graph G^2, the graph G^2_θ includes a new vertex D, that has only in-neighbors, and serves as a "drain".

Definition 3.4. [G^2_θ] Given a node-pair graph G^2 and a threshold $0 < \theta < 1$, $G^2_\theta = (V_\theta \cup \{D\}, E_\theta, W_\theta)$, where: $V_\theta \subseteq V^2$ is a set of nodes and D is a new node, $E_\theta \subseteq (V_\theta \cup \{D\} \times V_\theta \cup \{D\})$ is the edges set and W_θ is a weight function, defined as follows.

- **Nodes:** A node $(u,v) \in V_\theta$ iff $sem(u,v) > \theta$.
- **Edges:** An edge $e = ((u,u'),(v,v')) \in E_\theta$ iff at least one of the following conditions holds
  (1) The nodes $(u,u'),(v,v')$ are adjacent in G^2.
  (2) There exists a walk in G^2, where $w = \langle(u,u'), w_1, \ldots, w_k, (v,v')\rangle$ and the node-pairs $w_1 \ldots, w_k \notin V_\theta$.
- **Weights:** The weight of an edge $e = ((u,u'),(v,v'))$ is defined as $W_\theta(e) = W_1(e) + W_2(e)$ where: $W_1(e) = W_{G^2}(e)$ if $e \in G^2$ and $(u,u'),(v,v') \in V_\theta$ and 0 otherwise, and $W_2(e) = \sum_{w:(u,u')\rightsquigarrow(v,v')} P[w]\cdot c^{l(w)-1}$, where $t = \langle(u,u'), w_1, ..., w_k, (v,v')\rangle$ is a path in G^2 and the node-pairs $w_1, ..., w_k \notin V_\theta$.
- **Edges to D:** Edges to the vertex D are added as follows: $\forall (u,u') \in V_\theta$ if the sum of all out-edges of (u,u') in the graph G^2_θ is different then the sum of all out-edges of (u,u') in the graph G^2, then $((u,u'),D) \in E_\theta$ and $W_\theta((u,u'),D)$ is set to be the difference.

In the last point, to ensure that all weights are strictly positive, we can prove that for every node in G^2_θ, the sum of out-edges in the original graph G^2 is always $\geq$ than the sum of out-edges in the reduced graph G^2_θ. Additional edge pruning can be done by the removal of all out-edges from singleton nodes. Since only the first meeting point of the surfers affects similarity scores, such edges can be omitted without changing scores (proof omitted). For example, Figures 2b and 2c depict a partial graph G^2 and its reduced version G^2_θ (faded nodes are dropped).

The similarity scores over G^2_θ, denoted as $s_\theta(\cdot,\cdot)$, are defined as the result of the random surfing computation on the reduced graph. I.e., if $(u,v) \notin V_\theta$ then $s_\theta(u,v) = 0$, else: $s_\theta(u,v) = sem(u,v)\sum_{w:(u,v)\rightsquigarrow(x,x)} P[w]\cdot c^{l(w)}$, where w is a path in G^2_θ and $l(w)$ is its length. We can now provide an alternative way to compute SemSim scores over the graph G^2_θ.

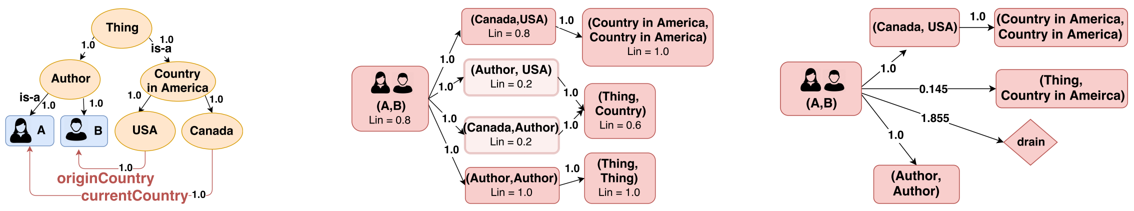
Theorem 3.5. $\forall (u,v) \in V_\theta : s_\theta(u,v) = sim(u,v)$

In conclusion, as we show in our experiments, the size of the graph G^2_θ is considerably smaller than that of G^2 and consequently, computing SemSim over G^2_θ requires exploring far less and shorter paths, hence it is more efficient. However, when considering very large graphs, even this compact representation might still be excessively large. To that end, in the next section, we present an alternative approach that simulates two random surfers directly over G.

4 APPROXIMATED SEMSIM

We next present an alternative approach for an efficient computation of SemSim, based on a solution originally proposed for

[2]In contrast to the meta-path approach [37] that restricts attention only to same-labels paths.



(a) Sample graph G. **(b) All paths from *(A,B)* in G^2.** **(c) All paths from *(A,B)* in G^2_θ.**

Figure 2: Example graph G, its reversed graph G^2 and its reduced version G^2_θ ($\theta = 0.3$, $c = 0.8$).

SimRank [9]. First, we prove that a naïve solution of simply replacing the underlying uniform distribution with the semantic aware distribution leads to a quadratic increase in the sample size. To overcome this, we employ *Importance Sampling*. As employing it still entails a computational overhead, we provide a complementary pruning technique that significantly speeds up computation while maintaining low error rates. We first recall SimRank optimizations while addressing the emerging challenges of their implementation in SemSim, then present our refined framework.

4.1 SimRank's Basic MC Framework

Suppose that we have two reverse walks w_1 and w_2 from the nodes $u, v \in V$, resp., and they first meet at the τ-th step. I.e., the τ-th steps of w_1 and w_2 are identical, but for any $l < \tau$ their l-th steps are different. If the walks do not meet, then $\tau \to \infty$. Given two random walks of length $k-1$, $w_1 = \langle u_1, ..., u_k\rangle$ and $w_2 = \langle v_1, ..., v_k\rangle$, let w denote their *coupled random walk*, where $w = \langle (u_1, v_1), ..., (u_k, v_k)\rangle$. It has been shown in [13] that $simrank(u, v) = \mathbb{E}[c^\tau]$. The authors of [9] suggested a *Monte Carlo* (MC) approximation framework, utilizing this equality, by sampling separated random walks, and approximating the similarity score using the average meeting distance. Specifically, to approximate SimRank, the framework precomputes a set of reversed random walks from each node in G, s.t (i) each set has n_w walks, and (ii) each walk is truncated at step t. Then the estimated SimRank score of u and v is defined as:

$$\frac{1}{n_w}\sum_{l=1}^{n_w} c^{\tau_l}$$

where τ_l denotes the step at which the two walks, sampled from u and v resp., first met, and ∞ otherwise. The space and preprocessing time complexities of this framework are both $O(n \cdot n_w \cdot t)$, and the method takes $O(n_w \cdot t)$ time to answer a single-pair SimRank query.

An important observation underlying SimRank's MC framework is the fact that the probability of a coupled random walk sampled from G^2, can be computed by simply multiplying the two marginal probabilities of the separate walks, sampled from G. Formally, given a coupled walk $w = \langle (u_1, v_1), ..., (u_k, v_k)\rangle$, its probability is:

$$Pr[w] = \prod_{i=1}^{k-1} \frac{1}{|O(u_i, v_i)|}$$

Considering its probability using the separated walks sampled from G, we get:

$$\prod_{i=1}^{k-1}\frac{1}{|O(u_i)|}\prod_{i=1}^{k-1}\frac{1}{|O(v_i)|} = \prod_{i=1}^{k-1}\frac{1}{|O(u_i)||O(v_i)|} = \prod_{i=1}^{k-1}\frac{1}{|O(u_i, v_i)|}$$

That is, in SimRank, one can simply sample walks from each node separately, directly from G, without materializing G^2. We will next show that this is not the case for SemSim, and present a refined sampling method for the SARWs.

4.2 Naïve MC framework for SemSim

Analogously, for SemSim we have: $sim(u, v) = sem(u, v) \cdot \mathbb{E}_P[c^\tau]$, where P is the semantic-aware probability. Note that when using the semantic-aware probability, one can no longer sample the walks separately. To account for the semantic similarity during the sampling process, one must consider *a pair of nodes* in each step. A naïve solution would be to generate a set of SARWs from every node-pair, then directly apply the MC framework. Namely, we can get an adjusted framework for SemSim with the same time complexity and error rate as in SimRank (because the time complexity depends on the number of walks from each pair of nodes, and this solution has the same number, n_w, of walks from each pair). However, in SimRank, the sampling set is of size $O(n_w \cdot t \cdot n)$, whereas this solution requires a much larger sampling set, i.e. $O(n_w \cdot t \cdot n^2)$ walks, as it samples n_w walks for each pair. To avoid this quadratic increase of data storage, we use importance sampling [10].

4.3 IS-based MC framework for SemSim

The core idea of our solution is to sample separate walks directly from G, using a different distribution than the "unknown" distribution P, then, apply importance sampling to estimate the desired similarity scores [10]. For completeness of this paper, we provide a short overview on the importance sampling technique, then present our adjusted framework.

Importance sampling is a general technique for estimating properties of a distribution while only having samples generated from a different one. For a single pair $u, v \in V$, we wish to estimate the expected value of $sem(u, v) \cdot c^{l(w)}$, where w is a coupled random walk drawn from P, i.e.,

$$\mathbb{E}_P[sem(u, v) \cdot c^{l(x)}] = sem(u, v) \cdot \sum P(w) \cdot c^{l(w)}$$

Given n_w samples $w_1, ..., w_{n_w}$ of coupled random walks drawn from P, an empirical estimate of $\mathbb{E}_P[sem(u, v) \cdot c^{l(w)}]$ is:

$$\mathbb{E}_P[sem(u, v) \cdot c^{l(w)}] \approx sem(u, v) \cdot \frac{1}{n_w}\sum_{i=1}^{n_w} c^{l(w_i)}$$

Using a simple manipulation we get:

$$\mathbb{E}_P[c^{l(w)}] = \sum \frac{P(w) \cdot Q(w) \cdot c^{l(w)}}{Q(w)} \approx \frac{1}{n_w}\sum_{i=1}^{n_w} c^{l(w_i)}\frac{P(w_i)}{Q(w_i)}$$

where Q is a distribution s.t $\forall w$ if $Q(w) = 0$ then $P(w) = 0$. Namely, we get an unbiased estimator of the function $c^{l(w)}$ under the distribution P, using samples drawn from Q. In our case, we can only sample separated walks from G (to avoid materializing G^2), while the desired distribution is defined over walks from G^2. Indeed, the expected value of the new estimator is equal to the desired one, that is, for every node-pair u, v we have:

$$sem(u, v) \cdot \mathbb{E}_Q[\frac{P(w) \cdot c^{l(w)}}{Q(w)}] = \qquad (4)$$
$$sem(u, v) \cdot \mathbb{E}_P[c^{l(w)}] = sim(u, v)$$

where w is a coupled random walk from u and v, P is the semantic-aware distribution and Q is the proposal distribution. Note that this equality holds for any choice of Q and $sem(\cdot,\cdot)$. Let $\hat{s}_Q(u,v)$ denote the score obtained using the MC simulation with a distribution Q. We can prove the following proposition, that ensures the approximation method has a bounded error (as was proven for SimRank [9]).

Proposition 4.1. *For a node-pair u, v, with at least $1-2e^{-n_w \cdot \frac{\epsilon^2}{2\cdot(1+\epsilon/3)}}$ probability: $|\mathbb{E}[\hat{s}_Q(u,v)] - \hat{s}_Q(u,v)| \leq \epsilon$, where n_w is the number of walks from each node and ϵ is the error rate.*

However, we note that $\mathbb{E}[\hat{s}_Q(u,v)] \neq sim(u,v)$, due to the truncation imposed on the sampled walks. To address this issue, following the analysis provided in [39] we get:

$$|\mathbb{E}[sim(u,v)] - \hat{s}_Q(u,v)| =$$

$$|\mathbb{E}[sem(u,v)\cdot c^{\tau}] - sem(u,v)\cdot Pr[\tau \leq t]\mathbb{E}[c^{\tau}|\tau \leq t]| =$$

$$sem(u,v)\cdot |Pr[\tau > t]\cdot \mathbb{E}[c^{\tau}|\tau > t]| \leq sem(u,v)\cdot c^{t+1} \leq c^{t+1}$$

By Prop. 4.1 and the inequality above, using union bound, we can prove the following.

Proposition 4.2. *For any node-pair u, v and $0 < \epsilon, \delta < 1$, if $t > log_c(\frac{\epsilon}{2})$ and $n_w \geq \frac{14}{3\epsilon^2}(log(\frac{2}{\delta}) + 2log(n))$, with at least $1-\delta$ probability: $|\hat{s}_Q(u,v) - sim(u,v)| \leq \epsilon$*

Furthermore, we can prove that the probability of interchanging two nodes in the similarity ranking of a node u converges to zero exponentially in the number of sampled walks n_w.

Proposition 4.3. *For every nodes u, v and v', such that $\delta = sim(u,v) - sim(u,v') > 0$ we have:*

$$Pr[\hat{s}_Q(u,v) < \hat{s}_Q(u,v')] \leq 2e^{-\frac{n_w\cdot\delta^2}{2+2\frac{\delta}{3}}}$$

Note, however, that the accuracy of estimation depends on the variance of the estimator, $Var(\hat{s}_Q(u,v))$, which in turn depends on the distribution Q. In general, $Var(\hat{s}_Q(\cdot,\cdot))$ is bounded in $[0,1]$, since the similarity scores are bounded in $[0,1]$. Therefore, we wish to find a distribution Q s.t. (i) the sampling process and probabilities computation can be done efficiently and (ii) $Var(\hat{s}_Q(\cdot,\cdot))$ is minimal. Here, since we do not have a-priori knowledge on either the semantic similarity or the meeting points of coupled walks, we choose Q to be the uniform distribution. See [27] for a discussion of other possible choices.

We are now ready to present Algorithm 1, an MC framework for computing single-pair SemSim scores, assuming, w.l.o.g., that Q is the uniform distribution. Ignore, for now, the lines highlighted in red. At preprocessing, we generate n_w random walks from each node, drawn from Q. Then, when a single-pair query arrives, $sim(u,v)$, we consider the set of coupled random walks starting from u and v. For each coupled walk, if the two walks indeed meet, the probability of their prefix until the meeting point is computed according to the distributions P and Q (lines 10 – 16). Then, the obtained score is added to the total similarity score (line 19). Finally, the estimated score is divided by the number of samples n_w (line 20).

Proposition 4.4. *For every $u, v \in V$, the expected output of Algorithm 1 is $sim(u,v)$, and the average time complexity is $O(n_w \cdot d^2 \cdot t)$, where n_w is the number of sampled walks from each node, t is their length and d is the average in-degree in the graph G.*

That is, an additional factor of d^2 is added to our framework's running-time. However, as we will show next, we can compensate for this by employing a pruning-based optimization.

Algorithm 1: IS-based MC framework for SemSim.

```
Input  : $n_w$ walks of length $t$ from each node, a decay
         factor $c$, and a threshold $\theta$.
1  $sim = 0$
2  if $sem(u,v) \leq \theta$ then
3      return 0
4  for $i = 1,...,n_w$ do
5      Let $w_i$ denote the coupled walk of the $i$-th walks from $u$ and
        $v$
6      Let $k$ be the samllest offset s.t the $i$-th walk from $u$ and from
        $v$ meet
7      if such $k$ exists then
8          Let $\tau(w_i)$ denote the prefix of $w_i$ up to offset $k$
9          Denote $\tau(w_i) = \langle(u_1,v_1),\ldots,(u_k,v_k)\rangle$, $Pw = 1$,
            $Q_W = 1$, $sim_w = 1$.
10         for $i = 1,\ldots,k-1$ do
11             $Pw\cdot = sem(u_{i+1},v_{i+1})\cdot W(u_{i+1},u_i)\cdot W(v_{i+1},v_i)$,
                $SO = 0$
12             for $I_j(u_i)$ in $I(u_i)$ do
13                 for $I_z(v_i)$ in $I(v_i)$ do
14                     $SO+ = W(I_j(u_i),u_i)\cdot W(I_z(v_i),v_i)\cdot$
                        $sem(I_j(u_i),I_z(v_i))$
15             $Pw/= SO$, $Q_W\cdot = |I(u_i)|\cdot|I(v_i)|$
16             $sim_w = sim_w \cdot \frac{Pw}{Q_W}\cdot c$
17             if $sim_w \leq \theta$ then
18                 break
19         $sim = sim + sim_w$
20 return $\frac{sem(u,v)\cdot sim}{n_w}$
```

4.4 Pruning

Similarly to the idea presented for G^2, we provide a pruning technique for the MC framework, avoiding the similarity computation of low probability walks. Recall that for G^2_θ the suggested technique ensures that scores above a given threshold would not be effected. Here, the approximation error may increase up to a controlled threshold. While in G^2 it is good pruning-wise to use high thresholds, here, if we use too high value the error grows and the scores may become meaningless, thus lower values are advisable. As opposed to the unbiased estimator we devised in the previous section, our pruning technique adds a one-sided additive error to scores. However, as we demonstrate in our experiments, it accelerates the performance significantly, while successfully distinguishing highly similar pairs from the rest.

For a coupled random walk w, let $s(w)$ denote the contribution of w to the similarity score. Since in every estimation we consider n_w coupled walks, it follows that:

$$s(w) = \frac{1}{n_w}\cdot\frac{P[w]\cdot c^{(l(w))}}{Q[w]}$$

Instead of computing the exact score of $s(w)$, our idea is to upper bound it. Concretely, given a coupled walk $w = \langle w_1,..,w_k\rangle$, with a closer look on $s(w)$ we get:

$$s(w) = \prod_{i=1}^{k-1}\frac{P[w_i \to w_{i+1}]\cdot c}{Q[w_i \to w_{i+1}]} \leq \prod_{i=1}^{l}\frac{P[w_i \to w_{i+1}]\cdot c}{Q[w_i \to w_{i+1}]}$$

where $0 < l < k-1$. Namely, $s(w)$ can be bounded in each step in the chain. Therefore, given a threshold θ, we can avoid computing the exact score, if in a certain step, the obtained score is smaller than the bound θ, as from this step on the score can only decrease. Formally, given a threshold $\theta \in [0,1]$ and a coupled walk w, we define $\hat{s}(w)$ as follows:

Table 2: Datasets.

Dataset	Size	Tasks
AMiner	$\lvert V\rvert = 0.35M$ $\lvert E\rvert = 3M$	Entity Resolution
Amazon	$\lvert V\rvert = 0.6M$ $\lvert E\rvert = 6M$	Link Prediction
Wikipedia	$\lvert V\rvert = 4.7K$ $\lvert E\rvert = 101K$	Relatedness
WorNet	$\lvert V\rvert = 82K$ $\lvert E\rvert = 128K$	Relatedness

Definition 4.5 (Approximated coupled walk score). The approximated score of a coupled walk w is defined as:

$$\hat{s}(w) := \prod_{i=1}^{l} \frac{P[w_i \to w_{i+1}] \cdot c}{Q[w_i \to w_{i+1}]} \leq \theta$$

where l is the smallest index this equation holds. If no such l exists, then $\hat{s}(w) = s(w)$.

As in the pruning done for G^2_θ, we can avoid computing scores of all pairs u, v s.t. $sem(u, v) < \theta$ (and again, obtain an error bounded by θ). In such cases, the result score is set to 0.

Consider again Algorithm 1. The highlighted red lines indicate our pruning refinements. In particular, in lines $2-3$, similarity scores of node-pairs with low semantic scores are set to 0, and in lines $17-18$ we ensure that scores of coupled walks are above θ, and otherwise, are bounded. We can prove that given $\theta \in [0, 1]$, the additional additive error is bounded by θ.

Proposition 4.6. *Given $\theta \in [0, 1]$, the additional additive error of the IS-based MC framework with pruning is bounded by θ.*

To ensure the estimated scores lies in $[0, 1]$, we add the following constraint on θ.

Lemma 4.7. *For every $\theta \in [0, 1-c]$ and $u, v \in V$, the approximated similarity score $\hat{s}_Q(u, v)$ obtained by Algorithm 1 with pruning lies in $[0, 1]$, where $c \in [0, 1)$.*

This lemma implies that the MC framework with pruning can efficiently capture big differences among similarity scores. But when it comes to small differences, the error of approximation obscures the actual similarity ranking, and an almost arbitrary reordering is possible. However, for many similarity search applications it is sufficient to distinguish between very similar, modestly similar, and dissimilar nodes. In terms of run-time, while the worst-case time complexity remains the same (no pruning is done), our experiments show pruning to be extremely effective in practice, yielding running times on par with SimRank.

Concluding, as mentioned in the Introduction, multiple optimizations techniques have been developed for SimRank based on SimRank's MC framework. Our framework extends for them as well. We discuss this in more details in our technical report [27], providing several examples, and also demonstrate this in our experimental study.

5 EXPERIMENTAL RESULTS

We complement our work with an experimental study, conducted to examine the performance of our measure as well as its usefulness in capturing objects' similarity, compared to measures proposed in previous work.

5.1 Experimental setup

We implemented SemSim in Java 7, and demonstrate its performance using Lin as the integrated semantic measure. All experiments were conducted on a Linux machine with 96GB of memory. We next describe the graph datasets we examined, then detail the parameters setting.

Datasets. We used several graph datasets, commonly used in the literature, which suitably include objects possessing both structural information and semantic meaning, as depicted in Table 2. Unless stated otherwise, all edge weights were set to 1.

AMiner. This graph was extracted from [1], and contains data on 1.5M academic papers. We extracted a weighted co-author graph focused on 30 database conferences. From each paper, we extracted its authors and relevant terms. In addition, we incorporated a domain taxonomy, built by aligning the terms with concepts from DBpedia [5]. The graph includes edges of three types: (1) collaboration edges (with weights reflecting the number of collaborations between two authors); (2) terms-authors edges (where weights correspond to the prevalence of the term in a given author's papers) and (3) taxonomy edges.

Amazon. This dataset was obtained from [18]. It contains 0.5M items from different categories, a domain taxonomy (obtained from Amazon product categorization), and information about co-purchased items. The edge types are: (1) edges between co-purchase items (with weights reflecting the number of times two items were bought together) and (2) taxonomy edges, linking between products to their categories, as well as categories to their super-categories.

Wikipedia. This dataset, obtained from [18], contains 4.7K Wikipedia articles, each is represented by a node. The domain taxonomy was built from Wikipedia categories. The edge types are: (1) links between articles and (2) taxonomy edges.

WordNet. This dataset is the noun sub-part of the lexical base WordNet [26]. The edge types are:(1) part-of relations, the non-hierarchical relations in WordNet and (2) taxonomy edges.

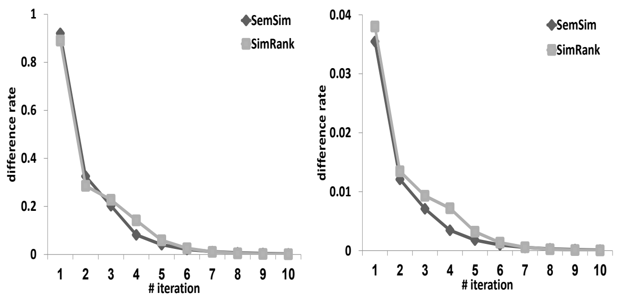
For both AMiner and Amazon datasets, we extracted a smaller versions to be used in the execution of the costly iterative forms of SemSim and SimRank. In AMiner, the small version includes the top 7K authors by number of publications, and in Amazon it includes the top 5K most bought items.

Parameter setting. For all datasets, we found the upper bound on the decay factor c by iterating on all node-pairs u, v computing $N_{u,v}$. We report that in all cases this value was > 0.6, a commonly used value for the decay factor in SimRank [24, 39]. Unless mentioned otherwise, for both SemSim and SimRank we used the following system parameters: The decay factor (c) was set to 0.6, and for the probabilistic framework, a set of 150 random walks of length 15 was sampled from each node. As for the threshold parameter θ used for pruning, we set $\theta = 0.05$. According to our experiments, this choice of the parameters allows for fast execution times, while maintaining negligible error rate.

5.2 Performance Evaluation

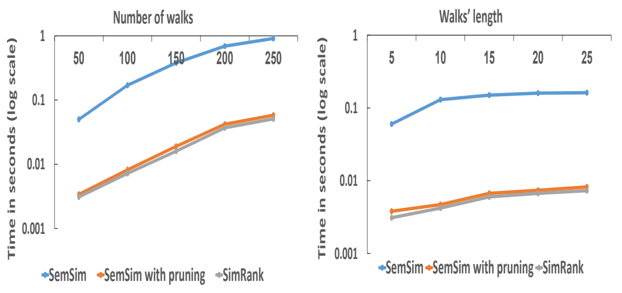
We review the performance of SemSim from five aspects: The convergence rate of its iterative form, the size of the reduced graph G^2_θ compared to the full graph G^2, the performance of Algorithm 1, in terms of execution times and error rate and the preprocessing phase.

Convergence. Our experiments validate empirically Prop. 2.4, showing that SemSim converges as fast as, and even faster than SimRank. We measured the average relative and absolute differences between similarity scores at consecutive iterations, for both SemSim and SimRank iterative forms, on different datasets. Results are depicted in Figure 3. Indeed, SemSim converged faster



(a) Avg. relative difference (b) Avg. absolute difference

Figure 3: Scores differences in consecutive iterations.



(a) t is set to 15 (b) n_w is set to 150

Figure 4: Average running times for single-pair similarity query.

Dataset		G^2	$G^2_\theta, \theta = 0.90$	$G^2_\theta, \theta = 0.95$
AMiner	# nodes	60M	14K	9K
	#edges	2.2B	39M	7.8M
	Avg. # of paths to singletons	17	10	5
	Avg. paths' length	4	3	3
Wikipedia	# nodes	22M	10K	6K
	#edges	10.2B	23.5M	4.7M
	Avg. # of paths to singletons	19	10	6
	Avg. paths' length	5	4	3

Table 3: The size of G^2 and G^2_θ for $\theta = 0.9$ (top ≈ 5K) or $\theta = 0.95$ (top ≈ 1K).

Dataset		SemSim with pruning $\theta = 0.05$	SemSim	SimRank
AMiner	Pearson's r	0.89	0.91	0.92
	Mean var	0.001	0.001	0.0004
	Max var	0.025	0.027	0.004
	Mean rel. err	0.405	0.397	0.274
	Max rel. err	0.478	0.468	0.364
	Mean abs. err	0.063	0.019	0.018
	Max abs. err	0.080	0.035	0.029
Amazon	Pearson's r	0.92	0.93	0.93
	Mean var	0.001	0.001	0.0005
	Max var	0.022	0.021	0.006
	Mean rel. err	0.366	0.320	0.298
	Max rel. err	0.428	0.399	0.389
	Mean abs. err	0.056	0.020	0.020
	Max abs. err	0.075	0.027	0.025

Table 4: Accuracy of approximation.

than SimRank, and in all experiments it converged after 5 iteration, i.e., the average (relative and absolute) difference between similarity scores was smaller than 10¯3.

Size of G^2_θ. We analyze the effectiveness of the G^2 pruning, demonstrating that when only highly similar objects are of interest (e.g., top-k queries), pruning with high θ values (e.g., 0.9, 0.95) is highly effective, and reduces the size of G^2 significantly. We refer the reader to Table 3, depicting a detailed comparison between the reduced graph G^2_θ (while setting $\theta = 0.9$ and $\theta = 0.95$), and the original graph G^2, constructed from the Amazon and Wikipedia datasets. In addition to the significant reduction in the number of nodes and edges, one can see that the average length of a path and the number of paths leading to singleton nodes (i.e., the number of paths that are considered while computing SemSim) were greatly reduced as well. However, while the measured size of G^2_θ is smaller than G^2 in approximately 3 orders of magnitude, this approach does not trivially scale for immense networks, in which the approximated framework is preferable.

Execution Times. We examine the running time of our MC framework (with and without pruning) as a function of the number of walks, n_w, and the truncation point t, compared to SimRank MC framework. Figure 4 depicts the average measured running times on the Amazon dataset (the results obtained over the other dataset demonstrated similar trends, and thus omitted from presentation). Not surprisingly, without pruning, the average time of SemSim is indeed slower than of SimRank: 0.217 ms and 0.0035 ms for SemSim and SimRank, resp. However, the running times with pruning are significantly faster, becoming close to those of SimRank (in average 0.0038 ms, where $\theta = 0.05$). Additionally, we examined the performance of both measures, using SLING optimization [39] recently suggested for SimRank (described in our technical report [27]), while storing probabilities only for node-pairs with semantic similarity scores >= 0.1. For both measures we achieved a significant improvement in times - 0.00021 ms and 0.00023 ms for SimRank and SemSim resp., with a memory consumption induced by the SLING index of size 1.1GB and 3.2GB, resp.

Approximation Error. We compared SemSim and SimRank approximated scores to scores computed by their iterative forms, to asses the cost of incorporating semantics. We then evaluated the error of approximation in terms of Pearson's r-correlation (by comparing approximated scores to the ground truth), variance, and error size. We report the error of approximation as measured for the (larger versions of) Amazon and AMiner dataset. In each experiment, we randomly selected 1K node-pairs, and computed the approximated similarity scores in 100 runs (rebuilding the random walks index before each run). The results, provided in Table 4, depict the Pearson's r-correlation values (achieved by comparing the approximated scores to the ground truth ones, obtained by the iterative forms), the (mean and maximal) variance of the estimators, and the (mean and maximal, relative and absolute) errors, as obtained by SemSim (with or without pruning) and SimRank.

As expected, SemSim's mean error is slightly higher than that of SimRank, yet both are in the same order of magnitude (0.366, 0.32, 0.298 for SemSim with pruning, without pruning, and SimRank, resp.). As discussed in Section 5.2, while the error of approximation for SemSim (with and without pruning) is slightly higher than for SimRank, the number of interchanges between the approximated scores and ground truth ones, as measured by the Pearson r-correlation[3], is significantly low and essentially equivalent to SimRank's (with or without pruning). This positive

[3]The Pearson r-correlation measures the "strength" of the linear association between ground truth scores and approximated ones.

result indicates that applying IS does not cause far apart scores to interchange, while maintaining execution times essentially on-par with SimRank.

Preprocessing. We complement the running times experiments by providing details regarding preprocessing time and the space costs of our framework. The offline processing, in which random walks are sampled, took approximately 2.5 minutes (in average over all datasets). While the sampling procedure performed as in SimRank, SemSim requires an additional work due to the semantic similarity measure. Following [11], we processed the taxonomical subpart of the graphs to facilitate constant-time Lin semantic similarity computations at run time. In all cases, the processing time took less 10 minutes. For example: In the Amazon dataset, where the taxonomy contains 2.5M edges, this phase took approximately 7 minutes.

The memory consumption of SemSim's MC framework is prominently due to the random walk index[4]. Additional memory costs for SemSim were due to the Lin semantic measure: storing the IC values and the data structure that allows for a constant time similarity computation (as described in [11]). Overall, the storage size was varied between 5 − 9MB, for all datasets, depending on the size of the taxonomical subpart of the particular graph.

5.3 Quality Evaluation

We examine the usefulness of SemSim compared to an extensive set of alternative measures for assessing node similarity, demonstrating the utility of SemSim when used in typical tasks, in which a similarity measure is required.

We used the following baselines from three common approaches for similarity assessment:
I. Structural-based measures: *SimRank* [13], *SimRank++* [2], a weighted variant of SimRank which ignores semantic information, and *Panther* [43], a random-walks based measure which considers edge weights as well.
II. Semantic similarity measures: *Lin* measure [23], as described in Section 2.2.
III. Measures combining structural and semantic information:. First, we employed *LINE* [38], an ML, node embedding-based similarity measure which accounts for latent semantic relations among the graph nodes. This serves a representative example for the state-of-the-art approach for assessing node similarity. Additionally, we tested *PathSim* [37], a HIN-dedicated similarity measure, which considers edge labels, and *Relatedness* [25] a semantic-aware measure which considers the properties' relating concepts. Last, we employed the *Multiplication* and *Average* competitors, returning the product (resp., average) of independent structural and semantic scores obtained by SimRank and Lin. These measures serve as baselines to our approach that interweaves structure and semantics throughout a recursive computation.

These baselines were examined in typical tasks in which a similarity measure is required: *Term Relatedness*, a problem requiring a measure aware of both semantic and structural knowledge, (tested on Wikipedia and WordNet datasets); *Link Prediction*, in which we used the measures to predict co-purchases in Amazon dataset, and *Entity Resolution* was tested on AMiner dataset, to detect duplication of entities. A ground truth was defined for each task, used to quantitatively evaluate the effectiveness of each competitor.

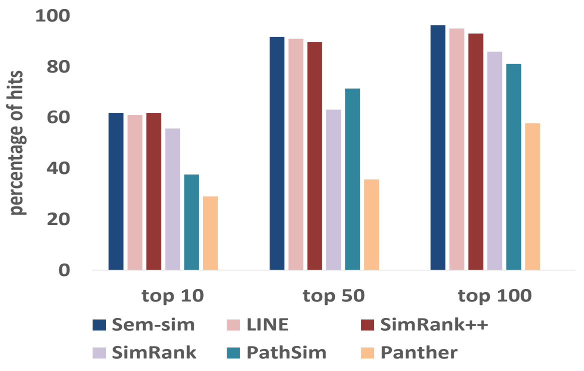
[4] Storage optimization techniques previously developed for SimRank can be directly applied in our settings as well (e.g. [14, 34]).

Method	r (Wiki)	p (Wiki)	r (WN)	p (WN)
Panther	0.323	0.0376	0.206	10^{-3}
PathSim	0.293	0.0662	0.332	10^{-3}
Simrank	0.295	0.0641	0.397	10^{-4}
Simrank++	0.296	0.0644	0.395	10^{-4}
Average	0.36	0.0514	0.401	10^{-4}
Multiplication	0.37	0.0508	0.409	10^{-4}
Lin	0.485	0.0015	0.449	10^{-4}
LINE	0.493	0.0001	0.470	10^{-4}
Relatedness	0.510	0.0007	0.488	10^{-4}
SemSim	**0.585**	0.0001	**0.501**	10^{-4}

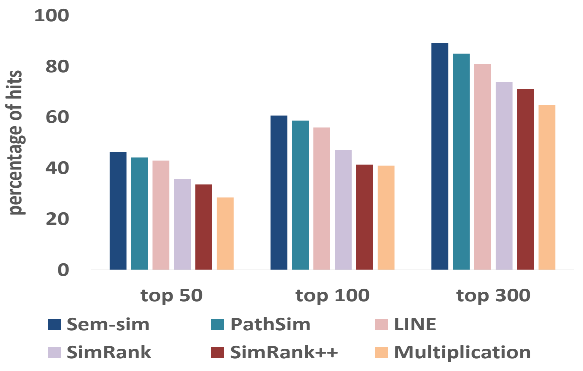
Table 5: Pearson's r and p-value in the WordsSim-test on Wikipedia (Wiki) and WordNet (WN).

Term Relatedness. Relatedness between terms is a well studied problem that requires a measure aware of both semantic and structural knowledge. To examine the adequacy of SemSim for capturing relatedness, we used two datasets that contain relations between terms: Wikipedia and WordNet. The ground truth was defined by the WordsSim-353 test [8], a public and commonly used benchmark containing pairs of words alongside their *relatedness* scores, as computed by people (e.g. "computer-keyboard" has the score of 0.76). We then compared the scores obtained by each competitor, using the Pearson correlation measure (a commonly used measure to evaluate the accuracy result for this benchmark [25]). We removed pairs of words that were missing in the graph from the benchmark, retaining 40 pairs for Wikipedia and 342 for WordNet. Table 5 depicts the results for all baselines. We note that other corpus-based designated methods were suggested for this task (e.g., [42]), but they require external sources besides the input graph, thus we did not include them in our benchmark. Observe that the structural based measures (e.g., SimRank, SimRank++ and Panther) demonstrate inferior results, as this task greatly relies on the semantic relations among the concepts. Furthermore, naïve semantic measures such as Lin, that perform a rather simplistic similarity comparison (i.e., rely only on the taxonomy "is-a" edges), surpassed both the structural measures, and the the *Average and Multiplication* competitors, yet were outperformed by LINE and the Relatedness measure, which better combine the structural and semantic aspects, and consider all edges in the graph. The Relatedness measure, designated specifically for this task, exceeded the ML-based measure LINE, yet interestingly, it was surpassed by SemSim.

Link Prediction. We next demonstrate how SemSim may be used to predict co-purchases in the Amazon dataset. To compare between different baselines, we omitted 7.5K edges between items from the original dataset, and examined how well the measures can be used to predict those missing links as follows: Given an endpoint of a removed link, we performed a top-k search to find similar nodes to the given endpoint. If, for a given measure, the returned k nodes include the pair endpoint, we counted a "hit", and otherwise a "miss". A similar idea was employed to evaluate similarity search in [43]. The results are depicted in Figure 5(a). For compactness, we omitted measures with particularly low scores. As opposed to the *Relatedness* task, this task relies mostly on structural knowledge, hence structural-based measures (e.g., SimRank++, Panther) outperformed the semantic-based ones (e.g., Lin). Here, LINE was able to outperform most competitors, yet SemSim managed to obtain a slight advantage,



(a) Link Prediction



(b) Entity resolution

Figure 5: Prediction in top-k

due to the additional semantic information it accounts for which LINE ignores (i.e., the IC values and the taxonomy relations).

Entity Resolution. Last, we used the similarity measures to identify multiple distinct entries representing the same author (e.g, *Susan B. Davidson* and *Susan Davidson*), or the same term (e.g, *Data structures* and *Data structure*). Using the Levenshtein string distance, we identified 30 pairs representing the same entry (24 term-pairs and 6 author-pairs), and used the baselines to predict duplicate nodes following similar lines to the link-prediction task (i.e., a top-k similarity search). The results are depicted in Figure 5(b), reporting for each baseline the precision in top k, for various values of k, (here again, measures with particularly low were omitted). First, note that the results for all baselines are not particularly high, since information commonly used for entity resolution (such as mail addresses, affiliations, string edit distance, etc.) was not included in the graph. As in the link-prediction task, the structural based measures outperformed the semantic based one (omitted from presentation). This stems from the particular characteristics of the AMiner graph, in which the semantic similarity of all authors nodes is identical (i.e., all authors nodes "is-a" *Author*). Hence, the semantic similarity between authors in this setting is not informative. Here again, the Multiplication/Average baselines demonstrate inferior results. It should be pointed out that PathSim outperforms most competitors, as it is a structural measure that also considers the edge labels, thus accounts for some semantics. However, SemSim managed to get an advantage, even if sometimes marginal, for all tested values of k.

In summery, as demonstrated, SemSim outperformed the competitors in all tasks mentioned above. In some tasks the advantage was marginal compared to the second-best for a specific task, yet we note that the second-best competitor was *different* in each task, illustrating the robustness of SemSim. For the rest of the baselines, results varied depending on the amount of semantics conveyed in the dataset, and of the degree to which the task is semantically complex. To conclude, the experiments indicate that even in cases where only partial semantics is available, SemSim serves as a robust measure and exploits this information to get an edge over the competitors.

6 RELATED WORK

SimRank is a popular measure and its potency was demonstrated in various scenarios [2, 6]. Several extensions that enrich it with more information (e.g. edge weights [2], particular paths [6] or second order walks [41]) were suggested, but they do not make full use of semantics available and thus, as illustrated in our experiments, yield less accurate estimations in semantic-sensitive tasks. Moreover, the optimization technique used in SimRank++ was build on *matrix multiplication* rather than random walks, and consequently, scalability issues were ignored. One of the contributions of this work is an efficient computation scheme, applicable also to several of these variants (e.g. [2, 45]).

As mentioned, a prominent body of work has focused on SimRank approximation techniques. These works are categorized into (a) matrix-based approaches [19], and (b) random walk-based approaches [14, 39]. A recent survey [44] advocates that the latter approach is more scalable, compatible with updates in the graph, and can be trivially parallelized. Therefore, we chose to extend it in accordance with our setting. As mentioned in Section **??**, the contributions of these works are applicable for SemSim, requiring only minor adaptations.

In our experiments we use Lin [23], a simple and effective *Information Content* (IC)-based semantic measure. However, as explained in Section 2, any other measure can be incorporated, given that it satisfies three intuitive constraints. Examples of other applicable semantic measures include: (i) IC-based measures [32], (ii) Edge-counting measures, which use the length of the shortest path between nodes in the estimation of similarity [31], and (iii) Feature-based measures [20, 42]. The former two regard a domain ontology, while the latter usually involves additional sources (e.g. textual corpus [20]).

Heterogeneous Information Network (HIN) is a ubiquitous model for real-world data, as it enables enriching simple graphs with additional useful information [36]. This, however, makes the assessment of node similarity challenging, as HIN paths convey latent semantic information. The majority of existing similarity measures for general networks do not consider all available information in the HIN. Specifically, measures such as [13, 43, 45] focus solely on the network structure, while measures suggested in [23, 32] concern only with the semantic information as implied by hierarchical relations. More recent works propose HIN-dedicated measures [35, 37], advocate considering only meaningful meta-paths between objects. But the choice of appropriate paths is made a-priori, and requires intimate knowledge of the dataset and the specific information needs[5] [22]. In contrast, SemSim is a generic measure that encompasses all available information, and automatically prioritizes meaningful paths.

As opposed to the declarative approach, recent work in the field of *representation learning* [4, 30] suggests *embedding* techniques that discover low-dimensional representations of graph nodes in a vector space. While this approach often outperforms

[5]Otherwise, an average of all paths can be taken, resulting in inferior results.

dedicated similarity measures for HINs (as demonstrated), a key disadvantage is that the results are hard to explain and interpret. Moreover, as we showed, SemSim not only retains its declarative definition but also yields more accurate similarity estimations in multiple tasks.

Incorporating semantic and structural information when determining relations between graph objects has also been proven useful in several related domains. Works in *ontology matching* and *entity resolution* suggest using both taxonomy edges and structural properties of nodes to properly align entities [12, 28]. However, their goal is different, as they aim to identify equivalent representations of the same entity, thus some of the core techniques employed (e.g., string matching) cannot be directly applied for measuring similarity between different objects. It would be interesting to investigate in future work whether SemSim can be employed in such contexts. An example domain where we showed such incorporation to be successful is similarity estimation for images that convey semantics, as we illustrated in [7], a demo that employed SemSim for the retrieval of similar Internet Memes.

7 CONCLUSION AND FUTURE WORK

In this paper we present SemSim, a similarity measure that refines SimRank with semantics, while preserving its intuitive definition and scalable computation. We introduce Semantic-Aware Random Walks, an extension of the traditional notion of random walks, that preserves properties necessary for applying existing SimRank's optimizations. Our probabilistic framework employs Importance Sampling along with an effective pruning technique, and maintains a negligible error rate. Our experiments further demonstrate the quality and robustness of SemSim in multiple practical scenarios, as well as the efficiency of our algorithms.

Several interesting directions are left for future research. First, in practice, information networks are often dynamic and may induce uncertainty, hence it would be important to extend SemSim to such settings. The use of parallelism and compact indexing mechanisms [3, 21], to achieve further computation speedup, are also an interesting direction for future work. Last, we have focused here only on single-pair queries. We intend on developing optimizations facilitating single-source and top-k similarity queries, inspired by [17, 46].

REFERENCES

[1] aminer [n. d.]. AMiner. https://aminer.org/data. ([n. d.]).

[2] Ioannis Antonellis, Hector Garcia Molina, and Chi Chao Chang. 2008. Simrank++: query rewriting through link analysis of the click graph. *PVLDB* (2008).

[3] Yuanzhe Cai, Gao Cong, Xu Jia, Hongyan Liu, Jun He, Jiaheng Lu, and Xiaoyong Du. 2009. Efficient algorithm for computing link-based similarity in real world networks. In *ICDM'09*. IEEE.

[4] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. 2015. Heterogeneous network embedding via deep architectures. In *SIGKDD*.

[5] dbpedia [n. d.]. DBpedia. http://dbpedia.org/About. ([n. d.]).

[6] Beate Dorow, Florian Laws, Lukas Michelbacher, Christian Scheible, and Jason Utt. 2009. A graph-theoretic algorithm for automatic extension of translation lexicons. In *GEMS*. Association for Computational Linguistics.

[7] Maya Ekron, Tova Milo, and Brit Youngmann. 2017. SimMeme: Semantic-Based Meme Search [Demonstration]. In *CIKM (CIKM '17)*.

[8] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *WWW*.

[9] Dániel Fogaras and Balázs Rácz. 2005. Scaling link-based similarity search. In *WWW*.

[10] Peter W Glynn and Donald L Iglehart. 1989. Importance sampling for stochastic simulations. *Management Science* (1989).

[11] Dov Harel and Robert Endre Tarjan. 1984. Fast algorithms for finding nearest common ancestors. *siam Journal on Computing* (1984).

[12] Wei Hu, Jianfeng Chen, and Yuzhong Qu. 2011. A Self-training Approach for Resolving Object Coreference on the Semantic Web. In *WWW (WWW '11)*.

[13] Glen Jeh and Jennifer Widom. 2002. SimRank: a measure of structural-context similarity. In *SIGKDD*.

[14] Minhao Jiang, Ada Wai-Chee Fu, and Raymond Chi-Wing Wong. 2017. READS: a random walk approach for efficient and accurate dynamic SimRank. *PVLDB* (2017).

[15] Mitsuru Kusumoto, Takanori Maehara, and Ken-ichi Kawarabayashi. 2014. Scalable similarity search for SimRank. In *SIGMOD*.

[16] Juan J Lastra-Díaz, Ana García-Serrano, Montserrat Batet, Miriam Fernández, and Fernando Chirigati. 2017. HESML: A scalable ontology-based semantic similarity measures library with a set of reproducible experiments and a replication dataset. *Information Systems* (2017).

[17] Pei Lee, Laks VS Lakshmanan, and Jeffrey Xu Yu. 2012. On top-k structural similarity search. In *ICDE*.

[18] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data. (2014).

[19] Cuiping Li, Jiawei Han, Guoming He, Xin Jin, Yizhou Sun, Yintao Yu, and Tianyi Wu. 2010. Fast computation of simrank for static and dynamic information networks. In *EDBT*.

[20] Yuhua Li, Zuhair A Bandar, and David McLean. 2003. An approach for measuring semantic similarity between words using multiple information sources. *TKDE* (2003).

[21] Zhenguo Li, Yixiang Fang, Qin Liu, Jiefeng Cheng, Reynold Cheng, and John Lui. 2015. Walking in the cloud: Parallel simrank at scale. *PVLDB* (2015).

[22] Jiongqian Liang, Deepak Ajwani, Patrick K Nicholson, Alessandra Sala, and Srinivasan Parthasarathy. 2016. What Links Alice and Bob?: Matching and Ranking Semantic Patterns in Heterogeneous Networks. In *WWW*.

[23] Dekang Lin. 1998. An information-theoretic definition of similarity.. In *ICML*.

[24] Dmitry Lizorkin, Pavel Velikhov, Maxim Grinev, and Denis Turdakov. 2008. Accuracy estimate and optimization techniques for simrank computation. *PVLDB* (2008).

[25] Laurent Mazuel and Nicolas Sabouret. 2008. Semantic relatedness measure using object properties in an ontology. In *ISWC*.

[26] George A Miller. 1995. WordNet: a lexical database for English. *ACM* (1995).

[27] Tova Milo, Amit Somech, and Brit Youngmann. 2018. Technical report. http://courses.cs.tau.ac.il/software1/1617b/misc/main-semsim-full.pdf. (2018).

[28] DuyHoa Ngo and Zohra Bellahsene. 2016. Overview of YAM++ -(not) Yet Another Matcher for ontology alignment task. *Web Semantics: Science, Services and Agents on the World Wide Web* (2016).

[29] Normalization [n. d.]. Wikipedia: Normalization. https://en.wikipedia.org/wiki/Normalization_(statistics). ([n. d.]).

[30] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*.

[31] Roy Rada, Hafedh Mili, Ellen Bicknell, and Maria Blettner. 1989. Development and application of a metric on semantic nets. *IEEE Transactions on systems, man, and cybernetics* (1989).

[32] Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007* (1995).

[33] Nuno Seco, Tony Veale, and Jer Hayes. 2004. An intrinsic information content metric for semantic similarity in WordNet. In *ECAI*.

[34] Yingxia Shao, Bin Cui, Lei Chen, Mingming Liu, and Xing Xie. 2015. An efficient similarity search framework for SimRank over large dynamic graphs. *PVLDB* (2015).

[35] Chuan Shi, Xiangnan Kong, Yue Huang, S Yu Philip, and Bin Wu. 2014. Hetesim: A general framework for relevance measure in heterogeneous networks. *TKDE* (2014).

[36] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2017. A survey of heterogeneous information network analysis. *TKDE* (2017).

[37] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *PVLDB* (2011).

[38] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*. WWW.

[39] Boyu Tian and Xiaokui Xiao. 2016. SLING: A near-optimal index structure for simrank. In *SIGMOD*.

[40] Wikipedia [n. d.]. Wikipedia: SimRank. https://en.wikipedia.org/wiki/SimRank. ([n. d.]).

[41] Yubao Wu, Yuchen Bian, and Xiang Zhang. 2016. Remember Where You Came from: On the Second-order Random Walk Based Proximity Measures. *PVLDB* (2016).

[42] Wen-tau Yih and Vahed Qazvinian. 2012. Measuring word relatedness using heterogeneous vector space models. In *NAACL HLT*.

[43] Jing Zhang, Jie Tang, Cong Ma, Hanghang Tong, Yu Jing, and Juanzi Li. 2015. Panther: Fast top-k similarity search on large networks. In *SIGKDD*.

[44] Zhipeng Zhang, Yingxia Shao, Bin Cui, and Ce Zhang. 2017. An Experimental Evaluation of SimRank-based Similarity Search Algorithms. *PVLDB* (2017).

[45] Peixiang Zhao, Jiawei Han, and Yizhou Sun. 2009. P-Rank: a comprehensive structural similarity measure over information networks. In *CIKM*.

[46] Weiguo Zheng, Lei Zou, Yansong Feng, Lei Chen, and Dongyan Zhao. 2013. Efficient simrank-based similarity join over large graphs. *PVLDB* (2013).