Unified and Incremental SimRank: Index-free Approximation with Scheduled Principle

(Extended Abstract)

Fanwei Zhu[†], Yuan Fang[#], Kai Zhang[†], Kevin Chen-Chuan Chang[⋆], Hongtai Cao[⋆], Zhen Jiang[†], Minghui Wu[†]

[†] Zhejiang University City College, China {zhufw, mhwu}@zucc.edu.cn
[#] Singapore Management University, Singapore yfang@smu.edu.sg
[⋆] University of Illinois at Urbana-Champaign, USA {kcchang, hongtai2}@illinois.edu

***Abstract*—SimRank is a popular link-based similarity measure on graphs. It enables a variety of applications with different modes of querying. In this paper, we propose UISim, a unified and incremental framework for all SimRank modes based on a scheduled approximation principle. UISim processes queries with incremental and prioritized exploration of the entire computation space, and thus allows flexible tradeoff of time and accuracy. On the other hand, it creates and shares common "building blocks" for online computation without relying on indexes, and thus is efficient to handle both static and dynamic graphs. Our experiments on various real-world graphs show that to achieve the same accuracy, UISim runs faster than its respective state-of-the-art baselines, and scales well on larger graphs.**

***Index Terms*—SimRank approximation, unification, index-free, scheduled principle, scalability**

I. Introduction

Graphs are ubiquitous nowadays, requiring effective similarity measures based on their link structures. SimRank is a popular link-based similarity measure on graphs based on the intuition " two objects are similar if they refer to similar objects"[1]. It enables a variety of applications with different modes of querying. However, its computation is prohibitively expensive even on a moderately large graph. In this paper, we study the efficient computation of different modes of SimRank problems (Table I).

TABLE I: SimRank Problems on a Graph $G = (V, E)$.

SimRank Problems		Query Q=(A,B)	Output
General definition	Partial-pair	$A \subseteq V$ $B \subseteq V$	$s(u, v)$: a A-by-B similarity matrix, with each entry $[S]_{u,v} = s(u, v)$
Popular modes	Single-pair	A={u} B={v}	$s(u, v)$: a single SimRank similarity score between u and v
	Single-source	A={u} B=V	$[S]_u$: a $\|V\|$-by-1 similarity vector, with each entry $[S]_{u,v} = s(u, v)$
	All-pair	A=V B=V	$[S]$: a $\|V\|$-by-$\|V\|$ similarity matrix, with each entry $[S]_{u,v} = s(u, v)$

Motivations. We study the existing SimRank approximation works and motivate our solution as follows:

National Science Foundation IIS 16-19302 and IIS 16-33755, Primary Research and Development Plan of Zhejiang Province 2021C01164, Zhejiang University ZJU Research 083650, Futurewei Technologies HF2017060011 and 094013, UIUC OVCR CCIL Planning Grant 434S34.

First, as there are distinct modes of SimRank for different scenarios, it is desirable to support all different modes in a unified manner by one algorithm for simplicity and robustness of system maintenance. In contrast, virtually all existing algorithms are designed for specific modes.

Second, as different applications may have specific requirement of the approximation, it is desirable to support flexible tradeoffs of efficiency and accuracy. In contrast, most other algorithms exhibit often a narrow range of tradeoff.

Third, as most real-world graphs are dynamic with frequent updates, it is desirable to support efficient online computation without relying indexes. In contrast, many other algorithms need to precompute and maintain an index to process online queries, and thus are not flexible to handle dynamic graphs.

Proposal. Motivated by these challenges, we propose a unified and incrementally-enhanced framework, UISim, to efficiently process different modes of SimRank queries. Specifically, *to support unification of different modes*, we investigate the query tours in different SimRank modes, and identify a "unified computation space of query tours" that is naturally adaptable to each distinct mode; *to support flexible tradeoff of time and accuracy*, we propose "a prioritized exploration of the computation space" to iteratively cover the query tours in an important-first manner; *for efficient computation without relying indexes*, we factorize the query tours into fine-grained segments that shared across iterations, and organize them as "building blocks" which can be easily computed and reused on-the-fly. We conduct extensive experiments to validate the superiority of UISim on graphs with different sizes and properties.

II. Summary of our approach

A. Unification of computation space

To support unified SimRank, we first investigate the computation space of SimRank problems. Conceptually, for any SimRank query $Q = (A, B)$, its computation space is composed by a set of *query tours* T_Q which can be assembled from necessary random walk tours starting from A and B, that is:

$$T_Q \equiv P_A \bowtie P_B = \{p_a \circ p_b \mid p_a \in P_A; p_b \in P_B\} \quad (1)$$

where $p_u \circ p_v \equiv u \leftsquigarrow x \rightsquigarrow v$ is the assembly of two same-length tours at the same meeting node x.

Thus, to calculate any similarity scores $S(Q)$, we can construct the set of corresponding query tours T_Q and process them in a unified framework– all tours in T_Q aggregate to the exact scores, while a subset of tours gives an approximation. Specifically, for a prioritized approximation, we discriminatively organize T_Q into disjoint subsets $T_Q = T_Q^0 \cup \cdots \cup T_Q^\eta$ such that tours in any partition T_Q^i are more important than tours in T_Q^{i+1}. Then, $S(Q)$ can be incrementally calculated through multiple iterations, with each iteration i computing a SimRank *increment* $\hat{S}^i(Q)$ over T_Q^i, adding up to the overall approximation $\hat{S}^{(i)}(Q) = \hat{S}^0(Q) + \cdots + \hat{S}^i(Q)$.

B. Benefit-based prioritized approximation

To process T_Q in an important-first manner, we propose to partition the partial tours P_A and P_B based on their importance, and schedule the assemblies based on their "benefit" (*i.e.*, the accuracy improvement) to the overall approximation. Specifically, given a set of hubs H (*i.e.*, high-degree nodes), the partial tours are first partitioned by their *hub length* $\mathcal{L}_h(p)$ (*i.e.*, number of hubs they pass through) as more hubs would significantly decay the importance of tours [2], that is, $P_u = P_u^0 \cup \ldots \cup P_u^\eta$ s.t.$\forall p \in P_u^i, \mathcal{L}_h(p) = i$. Next, for any two assemblies $A_{ij} : P_u^i \bowtie P_v^j$ and $A_{i'j'} : P_u^{i'} \bowtie P_v^{j'}$, we should schedule A_{ij} earlier than $A_{i'j'}$ ($A_{ij} \prec A_{i'j'}$), with the following criteria:

$$A_{ij} \prec A_{i'j'} \text{ if } \begin{cases} i + j \le i' + j' \\ \text{and} \\ |i - j| \le |i' - j'| \end{cases} \tag{2}$$

These criteria formalize the intuition that handling important tours would better improve the accuracy of estimation, and assembling symmetry partial tours would potentially generate more valid tours. They can be further integrated as $Max(i, j)$, indicating the priority of $P_u^i \bowtie P_v^j$ to be scheduled, since larger $i + j$ and larger $|i - j|$ would result in a larger $Max(i, j)$. Therefore, in iteration k, all the partial-tours assemblies $P_u^i \bowtie P_v^j$ with $Max(i, j) = k$ would be scheduled to generate the full tours, formalized as:

$$T_{(u,v)}^k = \bigcup_{Max(i,j)=k} P_u^i \bowtie P_v^j \tag{3}$$

C. Index-free online sharing of computation

To handle dynamic graphs, we further investigate how to efficiently realize the scheduled approximation without relying on any precomputed indexes.

First, to compute partial tours, we observe that partial tours $u \leftsquigarrow \ldots \leftsquigarrow h \leftsquigarrow v$ in P_u^i (*i.e.*, hub-length-i tours) can be obtained by extending the "prefix" tours $u \leftsquigarrow \ldots \leftsquigarrow h$ in P_u^{i-1} with the hub-length-0 "extension" tours $h \leftsquigarrow v$ at each hub. Thus, the reachability of hub-length-i tours can be efficiently computed by reusing $r^0(h|v)$ as building blocks:

$$r^i(u|v) = \sum_{h \in H_u^{i-1}} r^{i-1}(u|h) \cdot r^0(h|v) \tag{4}$$

where H_u^{i-1} is the border hubs of tours in P_u^{i-1}.

Second, to assemble two partial-tour sets, we can skip those "mis-matching" partial tours as they are not able to generate valid full tours. Accordingly, the aggregated reachability of full tours in $P_u^i \bowtie P_v^j$, can be obtained by assembling the reachability of length-matched partial tours that start at the same meeting node:

$$R\left(P_u^i \bowtie P_v^j\right) = \sum_{x \in X} \sum_{l \le M} \frac{1}{C^l} \left(r^{i,l}(u|x) \cdot r^{i,l}(v|x)\right) \tag{5}$$

where i,l denote hub length and natural length respectively, and M is the maximal natural length in computation.

III. Experimental Evaluation

We empirically evaluated UISim on eight real-world datasets with different properties and sizes, and compare UISim with the state-of-the-art competitors [3]-[6] in each query mode.

As all algorithms compute approximate SimRank scores, there is a trade-off between accuracy and query time. In order to fairly compare different methods, we vary the parameters of each method in a reasonably large range to to evaluate a large number of different configurations. We then compare their running time under the configurations that give similar accuracy. The experiments showed that UISim is substantially superior than previous state-of-the-art baselines in different modes (Fig. 1): 1) UISim always needs less time to achieve the same accuracy as its baseline, and 2) UISim achieves a good accuracy very fast while the baselines do not perform well within limited time. We also validate the scalability of UISim in growing graphs.

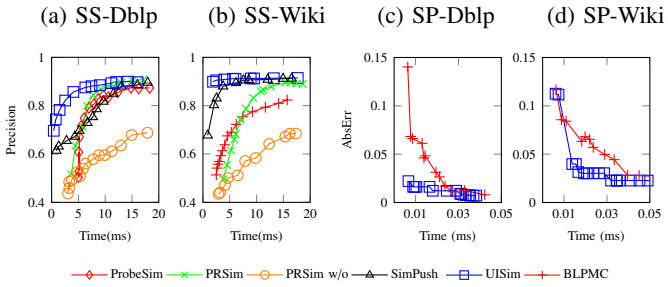

Fig. 1: Comparison with baselines in single pair (SP) and single source (SS) modes.

References

[1] G. Jeh and J. Widom. Scaling personalized web search. In WWW, pages 271–279, 2003.

[2] F. Zhu, Y. Fang, K. C.-C. Chang, and J. Ying. Scheduled approximation for personalized pagerank with utility-based hub selection. VLDBJ, pages1–25, 2015.

[3] Y. Liu, B. Zheng, X. He, Z. Wei, X. Xiao, K. Zheng, and J. Lu. Probesim: Scalable single-source and top-k simrank computations on dynamic graphs.PVLDB, 11(1):14–26, 2017.

[4] J. Shi, T. Jin, R. Yang, X. Xiao, and Y. Yang. Realtime index-free single source simrank processing on web-scale graphs. arXiv preprint arXiv:2002.08082, 2020.

[5] Z. Wei, X. He, X. Xiao, S. Wang, Y. Liu, X. Du, and J.-R. Wen. Prsim: Sublinear time simrank computation on large power-law graphs. In SIGMOD, pages 1042–1059, 2019.

[6] Y. Wang, L. Chen, Y. Che, and Q. Luo. Accelerating pairwise simrank estimation over static and dynamic graphs.PVLDB, 28(1):99–122, 2019.